

목 차

제1장 PLC의 기초 -----	1-1
1.1 PLC의 정의 및 적용 분야 기초 -----	1-1
1.1.1 PLC의 정의 -----	1-1
1.1.2 PLC의 적용 분야 -----	1-2
1.2 PLC의 구조 -----	1-2
1.2.1 PLC의 하드웨어 구조 -----	1-2
1.2.2 PLC의 소프트웨어 구조 -----	1-6
1. 3연산 처리 -----	1-9
1.4 PLC의 입출력 배선도 -----	1-11
1.4.1 PLC의 입력 배선도 -----	1-11
1.4.2 PLC의 출력 배선도 -----	1-14
1.5 PLC의 운전 모드 -----	1-18
제2장 GLOFA-GM 개요 -----	2-1
2.1 GLOFA-GM PLC의 특징 -----	2-1
2.1.1 IEC 표준 언어 -----	2-1
2.1.2 국제 규격의 통신 프로토콜 -----	2-2
2.1.3 윈도우 환경의 프로그램 TOOL(GMWIN)지원 -----	2-2
2.1.4 프로그램 작성 용이 -----	2-2
2.2 GLOFA-GM 성능 규격 -----	2-3
2.3 GLOFA-GM 제품 MAP -----	2-4
2.4 GLOFA-GM 시스템 구성 -----	2-4
2.4.1 GLOFA-GM7 시스템 구성 -----	2-4
2.4.2 12 슬롯 시스템 구성(GM4/6) -----	2-5
2.4.3 GM3/4 시스템 구성 -----	2-5
2.4.4 GM1 단독 CPU 시스템 구성 -----	2-6
2.4.5 GM1 멀티 CPU 시스템 구성 -----	2-6
2.4.6 GMR 이중화 시스템 구성(네트워크 이중화) -----	2-7
2.4.7 GMR 이중화 시스템 구성(중복 입출력) -----	2-8
2.4.8 GMR 이중화 시스템 구성(케이블 이중화) -----	2-9

목차

2.4.9 GLOFA-GM 리모트 시스템 구성-----	2-10
제 3 장 데이터 메모리 구성 -----	3-1
3.1 변수의 표현 방식 -----	3-1
3.1.1 직접 변수-----	3-1
3.1.2 네임드 변수-----	3-5
3.1.3 어레이 변수-----	3-9
제3장 프로그램 편집 TOOL(GMWIN) -----	4-1
4.1 기본 사용법-----	4-1
4.1.1 프로젝트 및 프로그램 정의-----	4-1
4.1.2 프로그램의 편집-----	4-4
4.1.3 컴파일 및 메이크-----	4-9
4.1.4 접속 및 전송-----	4-11
4.1.5 모드 전환 및 모니터-----	4-15
4.1.6 강제 I/O 설정-----	4-18
4.1.7 런 중 수정-----	4-20
4.1.8 시뮬레이터-----	4-21
4.2 화면 구성-----	4-22
4.2.1 메뉴-----	4-22
4.2.2 도구모음-----	4-28
4.2.3 도구상자-----	4-30
4.2.4 상태 표시 줄-----	4-30
4.3 프로젝트 구조-----	4-32
4.3.1 프로젝트-----	4-33
4.3.2 파라미터-----	4-37
4.3.3 라이브러리-----	4-42
제 5 장 프로그래밍 -----	5-1
5.1 시퀀스 프로그램 -----	5-1
5.1.1 시퀀스 연산자-----	5-1
5.1.2 입력 접점 및 출력 코일 프로그램-----	5-2
5.1.3 변환 검출 접점 및 변환 검출 코일-----	5-5
5.1.4 셋 코일 및 리셋 코일-----	5-9
5.2 평선 프로그램-----	5-11
5.2.1 평선과 평선 블록-----	5-11
5.2.2 기본 평선의 종류-----	5-12
5.2.3 기본 평선 프로그램-----	5-15
5.2.3.1 기본 평선 프로그램 작성-----	5-15
5.2.3.2 전송 평선-----	5-17

5.2.3.3 형 변환 평선-----	5-19
5.2.3.4 비교 평선-----	5-24
5.2.3.5 마스터 콘트롤-----	5-37
5.3 평선 블록 프로그램-----	5-40
5.3.1 타이머-----	5-40
5.3.1.1 TON-----	5-40
5.3.1.2 TOF-----	5-44
5.3.1.3 TP-----	5-47
5.3.2 응용 타이머 -----	5-52
5.3.2.1 TON_UINT -----	5-52
5.3.2.2 TOF_UINT-----	5-53
5.3.2.3 TMR -----	5-54
5.3.3 카운터-----	5-54
5.3.3.1 CTU -----	5-55
5.3.3.2 CTD -----	5-57
5.3.3.3 CTUD-----	5-59
5.3.4 SCON -----	5-61

부 록

- 부록 A. 표준 평선/평선블록 라이브러리
- 부록 B. 평선/평선블록 일람표
- 부록 C. 수치체계 및 데이터 구조
- 부록 D. 플래그 일람표
- 부록 E. 용어설명
- 부록 F. PLC 설치 환경 및 배선
- 부록 G. 유지 보수
- 부록 H. 자동화기기 제품 Map

제 1 장 PLC 의 기초

1.1 PLC 의 정의및 적용분야

1.1.1 PLC 의 정의

PLC(Programmable Logic Controller)란, 종래에 사용하던 제어반 내의 릴레이 타이머, 카운터 등의 기능을 LSI, 트랜스터 등의 반도체 소자로 대체시켜, 기본적인 시퀀스 제어 기능에 수치 연산 기능을 추가하여 프로그램 제어가 가능하도록한 자율성이 높은 제어 장치이다.

미국 전기 공업회 규격(NEMA: National Electrical Manufactrurers Association)에서는 “디지털 또는 아날로그 입출력 모듈을 통하여 로직, 시퀀싱, 타이밍, 카운팅, 연산과 같은 특수한 기능을 수행하기 위하여 프로그램 가능한 메모리를 사용하고 여러 종류의 기계나 프로세서를 제어하는 디지털 동작의 전자 장치”로 정의하고 있다.

1.1.2 PLC 의 적용 분야

설비의 자동화와 고 능률화의 요구에 따라 PLC 의 적용 범위는 확대 되고 있다. 특히 공장 자동화와 FMS(Flexible Manufacturing System)에 따른 PLC 의 요구는 과거 중규모 이상의 릴레이 제어반 대체 효과에서 현재 고기능화, 고속화의 추세로 소규모 공장 기계에서 대규모 시스템 설비에 이르기 까지 적용되고 있다.

표 1-1 은 PLC 제어 대상에 따른 적용 분야를 나타낸 것이다.

표 1-1

PLC 적용 분야

분 야	제 어 대 상
식료 산업	컨베이어 총괄 제어, 생산라인 자동 제어
제철, 제강 산업	작업장 하역 제어, 원료 수송 제어, 압연 라인 제어, 하역 운반 제
섬유, 화학공업	원료 수입 출하 제어, 직조 염색 라인 제어
자동차 산업	전송 라인 제어, 자동 조립 라인 제어, 도장 라인 제어, 용접기 제
기계 산업	산업용 로봇 제어, 공장 기계 제어, 송 · 배수 펌프 제어
상하수도	정수장 제어, 하수 처리 제어, 송 · 배수 펌프 제어
물류 산업	자동 창고 제어, 하역 설비 제어, 반송 라인 제어
공장 설비	압축기 제어
공해 방지사업	쓰레기 소각로 자동 제어, 공해 방지기 제어

1.2 PLC 의 구조

1.2.1 하드웨어 구조

(1) 전체구성

PLC 는 마이크로프로세서(microprocessor) 및 메모리를 중심으로 구성되어 인간의 두뇌 역할을 하는 중앙처리장치(CPU), 외부 기기와의 신호를 연결시켜 주는 입·출력부, 각 부에 전원을 공급하는 전원부, PLC 내의 메모리에 프로그램을 기록하는 주변 장치로 구성되어 있다.

그림 1-1 은 PLC 의 전체 구성도를 나타낸 것이다.

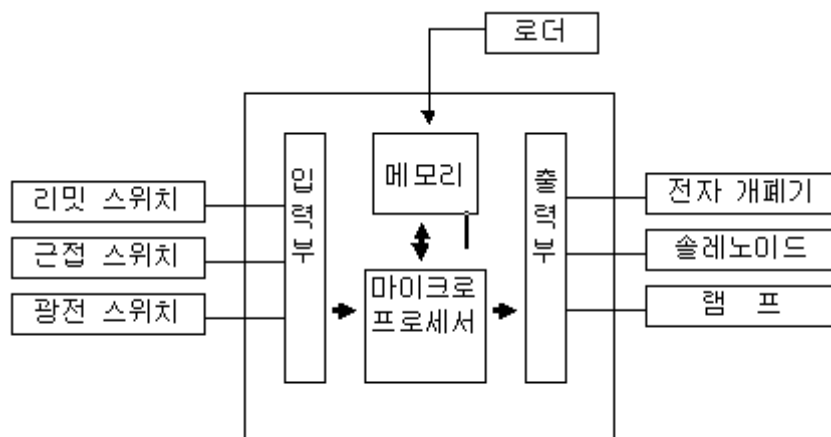


그림 1-1 PLC 의 전체 구성도

(2) PLC 의 CPU 연산부

PLC 의 두뇌에 해당하는 부분으로서 메모리에 저장되어 있는 프로그램을 해독하여 처리 내용을 실행한다. 이 절차는 매우 빠른 속도로 반복되며 모든 정보는 2 진수로 처리된다.

(3) PLC 의 CPU 메모리

① 메모리 소자의 종류

IC 메모리 종류에는 ROM(Read Only Memory)과 RAM(Random Access Memory)이 있으며 ROM 은 읽기 전용으로, 메모리 내용을 변경할 수 없다. 따라서, 고정된 정보를 써 넣는다. 이 영역의 정보는 전원이 끊어져도 기억 내용이 보존되는 불휘발성 메모리이다.

RAM 은 메모리에 정보를 수시로 읽고 쓰기가 가능하여 정보를 일시 저장하는 용도로 사용되나, 전원이 끊어지면 기억시킨 정보 내용을 상실하는 휘발성 메모리이다. 그러나 필요에 따라 RAM 영역 일부를 배터리 백업(battery back-up)에 의하여 불휘발성 영역으로 사용할 수 있다.

② 메모리 내용

PLC의 메모리는 사용자 프로그램 메모리, 데이터 메모리, 시스템 메모리 등의 3가지로 구분된다. 사용자 프로그램 메모리는 제어하고자 하는 시스템 규격에 따라 사용자가 작성한 프로그램이 저장되는 영역으로 제어 내용이 프로그램 완성 전이나 완성 후에도 바뀔 수 있으므로 RAM이 사용된다. 프로그램이 완성되어 고정이 되면 ROM에 써 넣어 ROM운전을 할 수 있다. 데이터 메모리는 입·출력 릴레이, 보조 릴레이, 타이머와 카운터의 접점 상태 및 설정값, 현재값 등의 정보가 저장되는 영역으로 정보가 수시로 바뀌므로 RAM영역이 사용된다.

시스템 메모리는 PLC 제작 회사에서 작성한 시스템 프로그램이 저장되는 영역이다. 이 시스템 프로그램은 PLC의 기능이나 성능을 결정하는 중요한 프로그램으로, PLC 제작 회사에서 직접 ROM에 써 넣는다.

(4) PLC의 입·출력부

PLC의 입·출력부는 현장의 외부 기기에 직접 접속하여 사용한다. PLC 내부에는 DC5(V)의 전원(TTL 레벨)을 사용하지만 입·출력부는 다른 전압 레벨을 사용하므로 PLC 내부와 입·출력의 접속(interface)은 시스템 안정에 결정적인 요소가 된다.

PLC의 입·출력부는 다음의 사항이 요구된다.

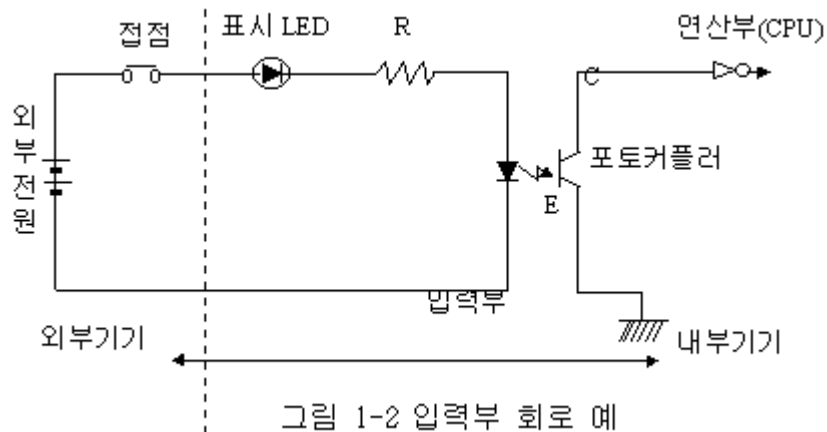
- ① 외부 기기와 전기적 규격이 일치해야 한다.
- ② 외부 기기로 부터의 노이즈가 CPU 쪽에 전달되지 않도록 해야 한다.
[포토 커플러(photocoupler) 사용]
- ③ 외부 기기와의 접속이 용이해야 한다.
- ④ 입출력의 각 접점 상태를 감시할 수 있어야 한다.(LED 부착) 입력부는 외부 기기의 상태를 검출하거나 조작 판넬을 통해 외부 장치의 움직임을 지시하고 출력부는 외부 기기를 움직이거나 상태를 표시한다.

표 1-2 입출력 기기

I/O	구 분	부 착 장
-----	-----	-------

I/O	구 분	부 착 장 소	외부 기기의 명칭
입력부	조작 입력	제어반과 조작반	푸시 버튼 스위치 선택 스위치 토글 스위치
	검출 입력 (센서)	기계 장치	리밋 스위치 광전 스위치 근접 스위치 레벨 스위치
출력부	표시 경보 출력	제어반 및 조작반	파일럿 램프 부저
	구동 출력 (액추에이터)	기계장치	전자 밸브 전자 클러치 전자 브레이크 전자 개폐기

외부 기기로부터의 신호를 CPU 의 연산부로 전달해 주는 역할을 한다. 입력의 종류로는 DC24[V], AC110[V] 등이 있고, 그 밖의 특수 입력 모듈로는 아날로그 입력(A/D) 모듈, 고속 카운터(high speed counter) 모듈 등이 있다.



나)출력부

내부 연산의 결과를 외부에 접속된 전자 접촉기나 솔레노이드에 전달하여 구동시키는 부분이다. 출력의 종류에는 릴레이 출력, 트랜지스터 출력, SSR(Solid State Relay)출력 등이 있고 그 밖의 출력 모듈로는 아날로그 출력(D/A) 모듈, 위치 결정 모듈등이 있다.

트랜지스터 출력부 회로의 예는 그림 1-3 과 같다.

출력 모듈을 출력 신호와 개폐 소자에 따라 분류하면 표 1-3 과 같다.

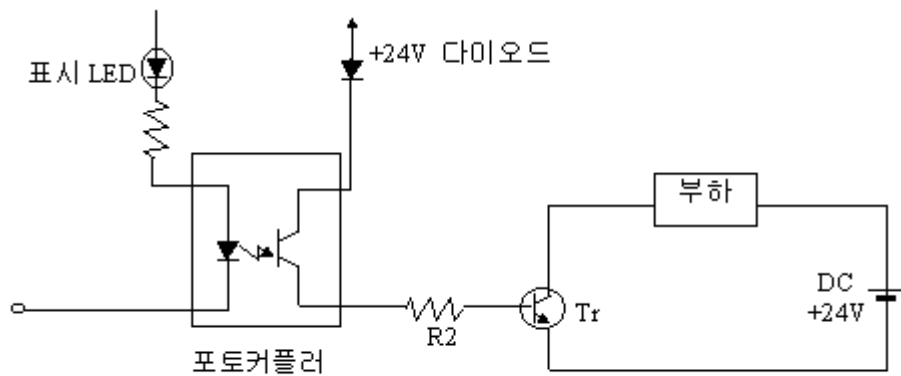


그림 1-3 트랜지스터의 출력부 회로

표 1-3 출력 모듈의 종류

출력 신호 종류	개 폐 소 자	
	유 점 점	무점점(반도체)
직류(DC)	릴레이 출력	트랜지스터 출력
교류(AC)	릴레이 출력	SSR 출력

표 1-3 에서와 같이 릴레이 출력은 직류나 교류를 모두 사용할 수 있으나 기계적 수명의 한계 때문에 점점의 개폐가 빈번할 경우는 교류 전원 전용인 무점점 SSR 출력이나 직류 전원 전용인 트랜지스터 출력을 사용하여야 한다.

1.2.2 소프트웨어 구조

(1) 하드 와이어드와 소프트웨어드

종래의 릴레이 제어 방식은 일의 순서를 회로도에 전개하여 그곳에 필요한 제어 기기를 절합하여 리드선으로 배선 작업을 해서 요구하는 동작을 실현한다. 이같은 방식을 하드와이어드 로직(hardwired logic)이라고 한다.

하드와이어드 로직 방식에서는 하드(기기)와 소프트웨어가 한쌍이 되어 있어 사양이 변경되면 하드와 소프트웨어를 모두 변경해야 하므로, 이것이 갖가지 문제를 발생시키는 원인이 된다. 따라서, 하드와 소프트웨어를 분리하는 연구 끝에 컴퓨터 방식이 개발되었다.

컴퓨터는 하드웨어(hardware)만으로는 동작할 수 없다. 하드웨어 속에 있는 기억 장치에 일의 순서를 넣어야만 비로소 기대되는 일을 할 수가 있다. 이 일의 순서를 프로그램이라 하며 기억 장치인 이 메모리에 일의 순서를 넣는 작업을 프로그래밍이라 한다.

이는 마치 배선작업과 같다고 생각하면 된다.

이 방식을 소프트웨어드로직(softwired logic)이라 하며 PLC 는 이 방식을 취하고 있다.

(2) 릴레이 시퀀스와 PLC 프로그램 차이점

PLC 는 LSI 등의 전자 부품의 집합으로 릴레이 시퀀스와 같은 접점이나 코일은 존재하지 않으며 접점이나 코일을 연결하는 동작은 소프트웨어로 처리되므로 실제로 눈에 보이는 것이 아니다.

또, 동작도 코일이 여자되면 접점이 닫혀 회로가 활성화되는 릴레이 시퀀스와는 달리 메모리에 프로그램을 기억시켜 놓고 순차적으로 내용을 읽어서 그 내용에 따라 동작하는 방식이다.

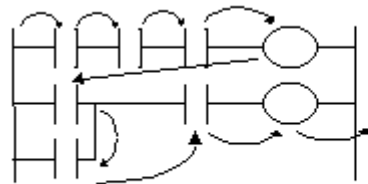
PLC 제어는 프로그램의 내용에 의하여 좌우된다.

따라서 사용자는 자유 자재로 원하는 제어를 할 수 있도록 프로그램의 작성 능력이 요구된다.

(가) 직렬 처리와 병렬 처리

PLC 시퀀스와 릴레이 시퀀스의 가장 근본적인 차이점은 그림 1-5 에 나타낸 것과 같이 “**직렬 처리**와 **병렬 처리**”라는 동작상의 차이에 있다.

PLC 는 메모리에 있는 프로그램을 순차적으로 연산하는 직렬 처리 방식이고 릴레이 시퀀스는 여러 회로가 전기적인 신호에 의해 동시에 동작하는 병렬 처리 방식이다. 따라서 PLC 는 어느 한 순간을 포착해 보면 한 가지 일 밖에 하지 않는다.

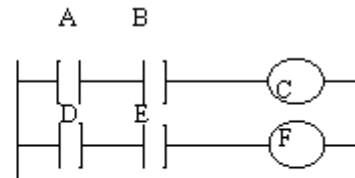


(a) 직렬 처리 방식

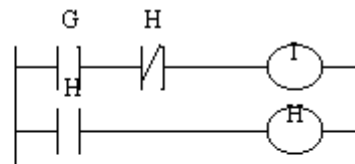


(b) 병렬 처리 방식

그림 1-5 연산 처리 방식



(a)



(b)

그림 1-6 시퀀스도

먼저 그림 1-6(a)의 시퀀스도로 PLC 와 릴레이의 동작상의 차이점을 설명한다. 릴레이 시퀀스에서는 전원이 투입되어 접점 A 와 B, 그리고 접점 D 와 E 가 동시에 닫히면, 출력 C 와 F 는 동작하고 어느 한쪽이 빠를수록 먼저 동작한다. 이에 비하면 PLC 는 연산 순서에 따라 C 가 먼저 출력되고 다음에 F 가 출력된다.

PLC 와 릴레이의 동작상의 차이점을 그림 1-6(b)의 경우에서 살펴 보면 먼저 릴레이 시퀀스에서는 전원이 투입되면 접점 J 가 닫힘과 동시에 H 가 동작되어 출력 I 는 동작될 수 없다.

PLC 는 직렬 연산 처리되므로 최초의 연산 때 G 가 닫히면 I 가 출력되고 J 가 닫히면 H 가 출력된다.

두 번째 연산 때 비로소 최초의 연산 때 출력된 H 에 의해서 I 의 출력이 해제된다.

(나) 사용 접점 수의 제한

릴레이는 일반적으로 개당 가질 수 있는 접점 수에 한계가 있다.

따라서 릴레이 시퀀스를 작성할 때에는 가능한 한 접점 수를 절약해야 한다.

이에 비하여 PLC 는 동일 접점에 대하여 사용 횟수에 제한을 받지 않는다.

이는 동일 접점에 대한 정보(ON/OFF)를 정해진 메모리에 저장해 놓고 연산할 때 메모리에 있는 정보를 읽어서 처리하기 때문이다.

(다) 접점이나 코일 위치의 제한

PLC 시퀀스에는 릴레이 시퀀스에는 없는 약속 사항이 있다.

그 하나는 코일 이후 접점을 금지하는 사항이다.

PLC 시퀀스에서는 코일을 반드시 오른쪽 모선에 붙여서 작성해야 한다.

그 밖에 PLC 시퀀스에서는 항상 신호가 왼쪽에서 오른쪽으로 전달되도록 구성 되어 있다.

따라서 PLC 시퀀스는 릴레이 시퀀스와는 다르게 오른쪽에서 왼쪽으로 흐르는 회로나, 상하로 흐르는 회로 구성을 금지하고 있다.

PLC 시퀀스의 약속 사항을 그림 1-7에 나타낸다.

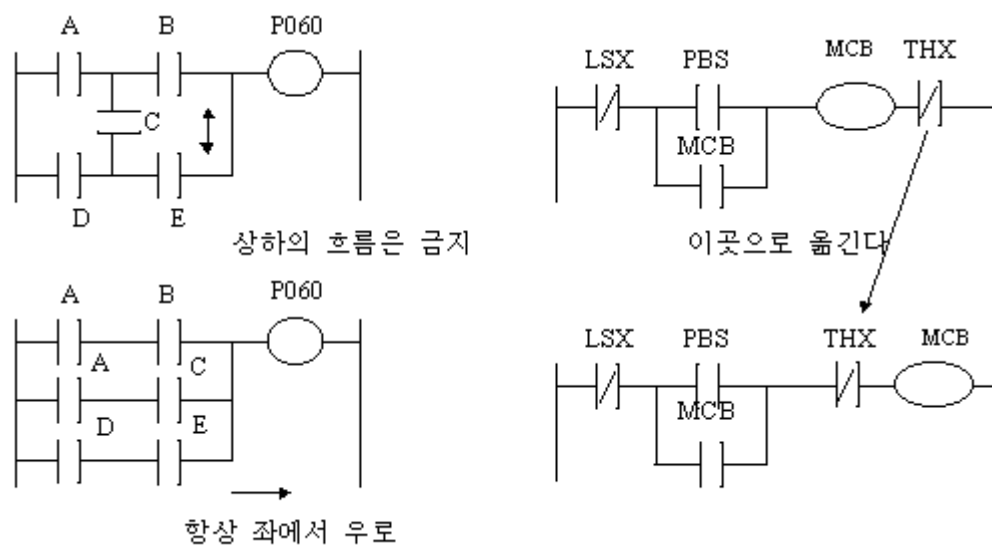


그림 1-7 PLC 시퀀스의 약속 사항

1.3 연산 처리

PLC의 연산 처리 방법은 입력 리프레시(Refresh) 과정을 통해 입력의 상태를 PLC의 CPU가 인식하고, 인식된 정보를 조건 또는 데이터로 이용하여 프로그램 처음부터 마지막까지 순차적으로 연산을 실행하고 출력 리프레시(Refresh)를 한다.

이러한 동작은 고속으로 반복되는데 이러한 방식을 ‘반복 연산 방식’이라 하고 한 바퀴를 실행하는데 걸리는 시간을 ‘1 스캔 타임’(1 연산 주기)라고 한다.

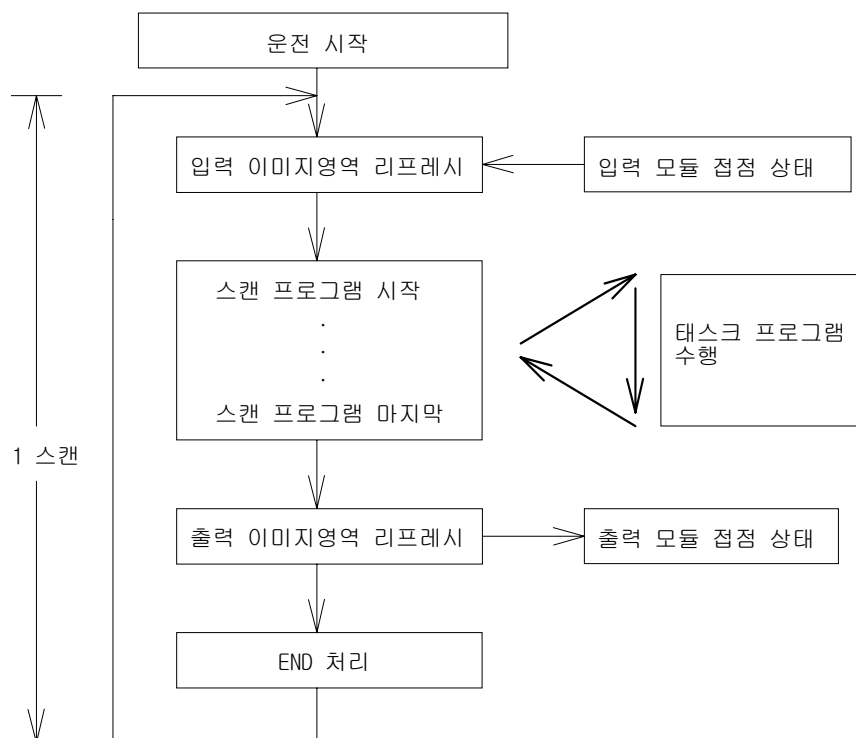


그림 1-8 PLC 연산 처리

① 입력 이미지 리프레시

PLC는 운전이 시작되면 입력 모듈을 통해 입력되는 정보들을 메모리의 입력 영역으로 받고, 이 정보들은 다시 입력 이미지 영역으로 복사되어 연산이 수행되는 동안의 입력 데이터로 이용된다. 이렇게 입력 영역의 데이터를 입력 이미지 영역으로 복사하는 것을 ‘입력 리프레시’(Input Refresh)라고 한다. 입력 리프레시는 운전이 시작될 때 뿐만 아니라 매 스캔 END 처리가 끝나면 그 순간의 입력 정보를 입력 이미지 영역으로 복사하여 연산의 기본 데이터 또는 연산의 조건으로 활용하게 된다.

② 프로그램 연산

입력 리프레시 과정에서 읽어 들인 입력 접점의 정보를 조건 또는 데이터로 이용하여 사전에 입력된 프로그램에 따라 연산을 수행하고 그 결과를 내부 메모리 또는 출력 메모리에 저장하게 된다.

GLOFA-GM PLC 에서 프로그램은 크게 스캔 프로그램과 태스크 프로그램의 두 가지로 나눌 수 있는데, 스캔 프로그램이란 PLC 의 CPU 가 RUN 상태면 무조건 수행하는 프로그램이고, 태스크 프로그램이란 특정 조건을 만족해야만 동작하는 프로그램이다.

스캔 프로그램 연산을 수행하는 도중에 태스크 프로그램의 실행 조건이 만족되면 스캔 프로그램의 연산을 멈추고, 태스크 프로그램을 수행한 후 태스크 프로그램으로 전이하기 직전에 연산이 수행되던 스캔 프로그램의 위치로 복귀하여 스캔 프로그램의 연산을 계속하게 된다.

③ 출력 리프레시

스캔 프로그램 및 태스크 프로그램의 연산 도중에 만들어진 결과는 바로 출력으로 보내어지지 않고 출력 이미지 영역에 저장되게 된다.

이 과정을 출력 이미지 리프레시라고 한다.

④ 자기 진단

연산의 과정에서 만들어진 결과는 바로 출력으로 내보내지 않고 출력 이미지 영역에 저장되게 된다. 그렇게 하는 이유는 프로그램의 마지막 스텝 연산이 끝나고 나면 PLC 의 CPU 는 시스템 상에 오류가 있는지를 검사하고 오류가 없을 때만 출력을 내보내기 때문이다.

만일 연산이 성공적으로 끝나서 그 결과가 출력 이미지 영역에 저장되었다고 해도 PLC 의 CPU 는 자기 시스템을 진단하여 시스템 상에 오류가 있다면 출력을 내보내지 않고 에러 메시지를 발생시키게 된다.

이것을 자기 진단이라고 한다.

⑤ END 처리

연산이 성공적으로 수행되고 자기 진단 결과 시스템에 오류가 없으면 출력 이미지 영역에 저장된 데이터를 출력 영역으로 복사함으로써 실질적인 출력을 내보내게 된다.

이 과정을 END 처리라 하며 END 처리가 끝나면 다시 입력 리프레시를 실시함으로써 PLC 는 반복적인 연산을 수행하게 된다.

1.4 PLC의 입출력 배선도

1.4.1 PLC의 입력 배선도

① DC 입력 배선(Sink/Source 타입:G4I-D24A)

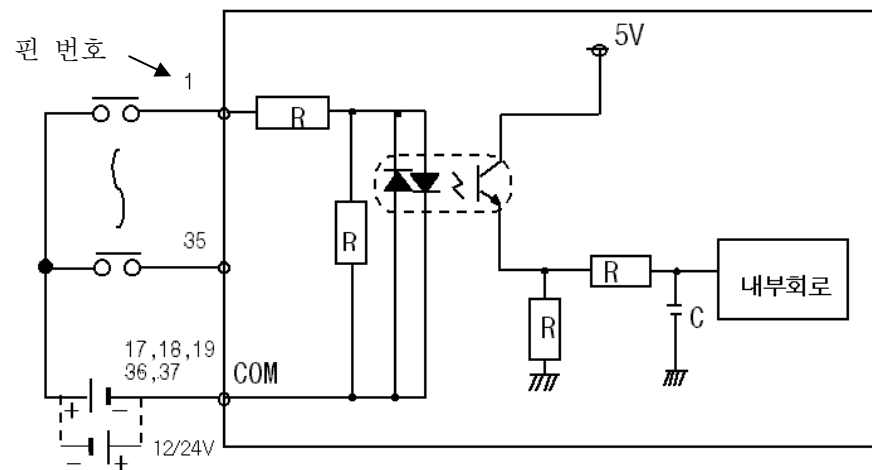


그림 1-9 DC 입력 모듈의 내부 회로(Sink/Source 타입)

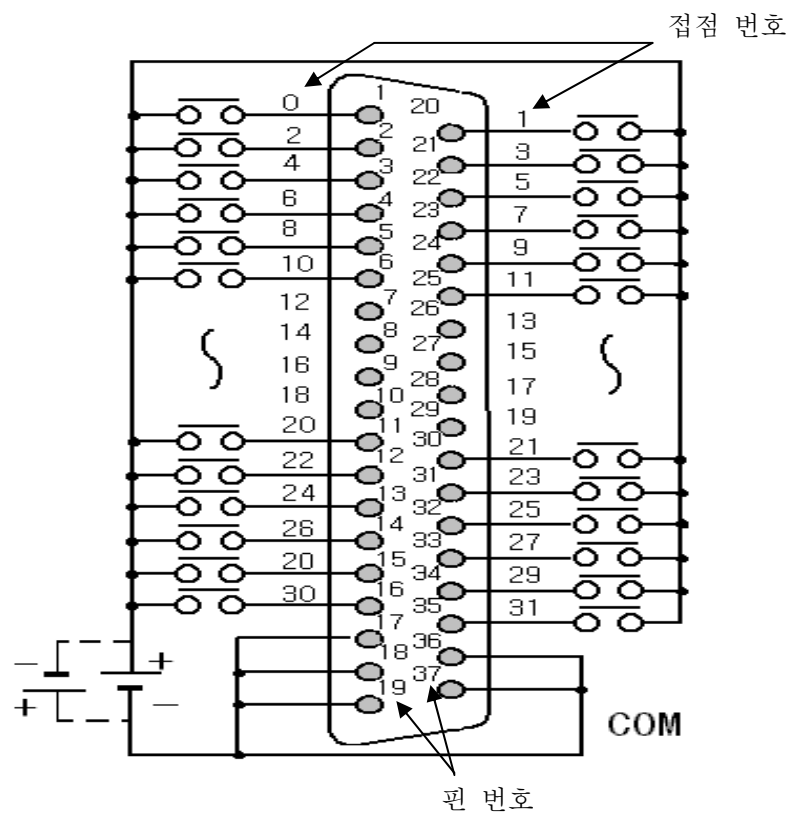


그림 1-10 G4I-D24A 외부 결선도

② DC 입력 배선 (Source 타입:G4I-D24B)

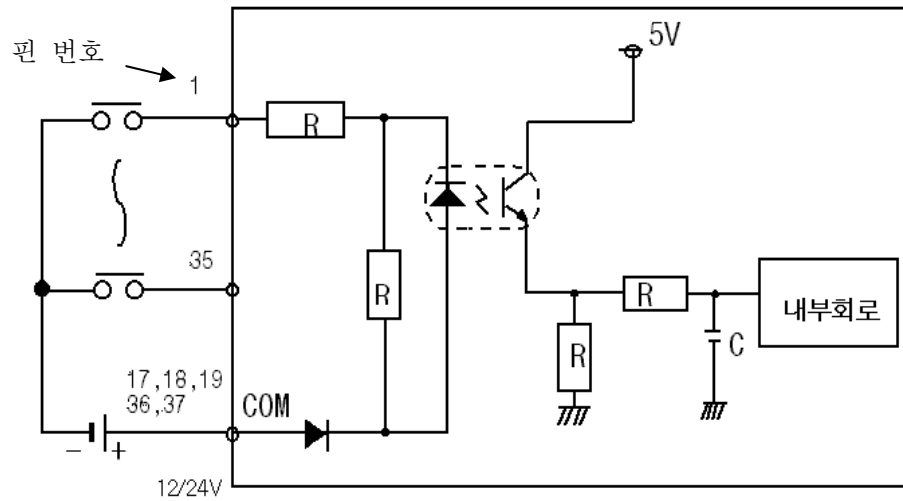


그림 1-11 DC 입력 모듈의 내부 회로(Source 타입)

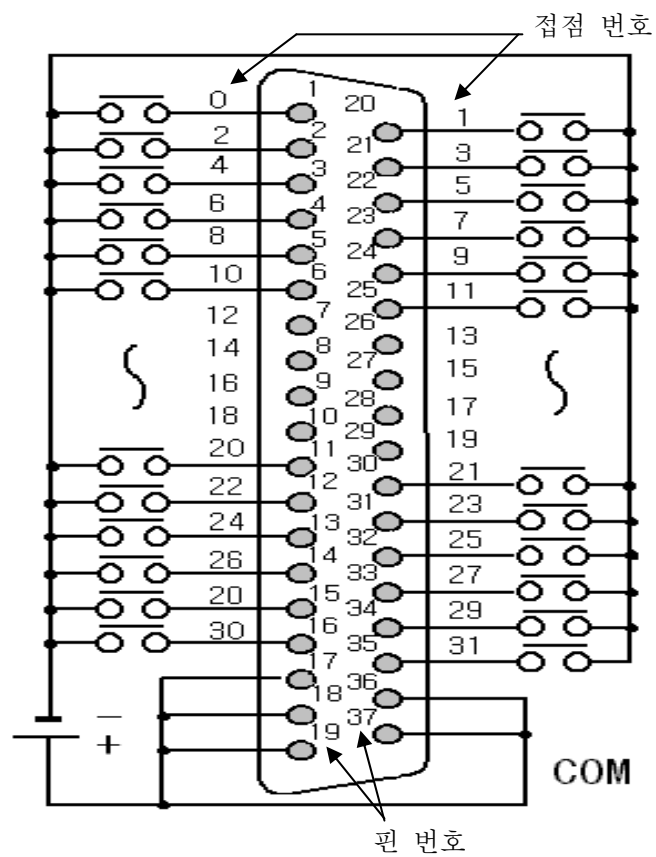


그림 1-12 G4I-D24B 외부 결선도

③ AC 입력 배선(G4I-A12A)

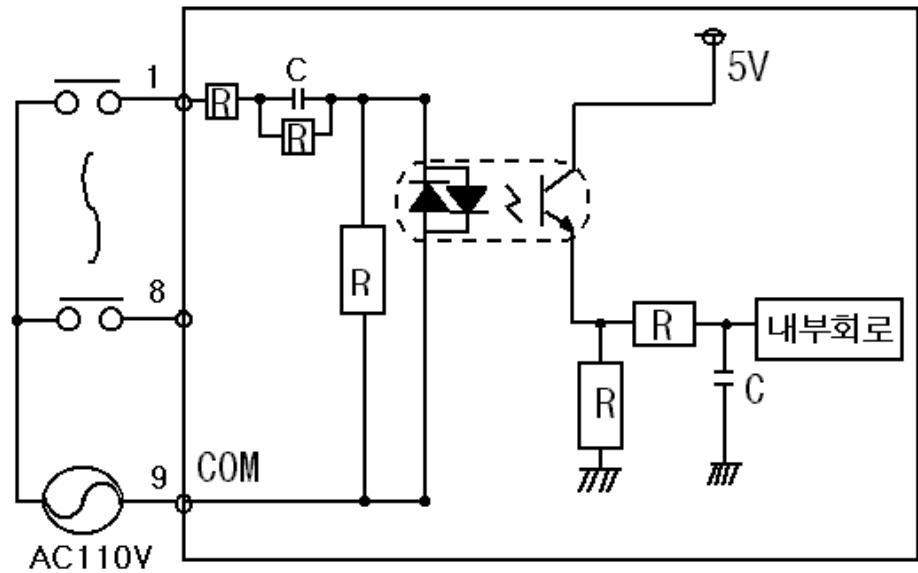


그림 1-13 AC 입력 모듈의 내부 회로

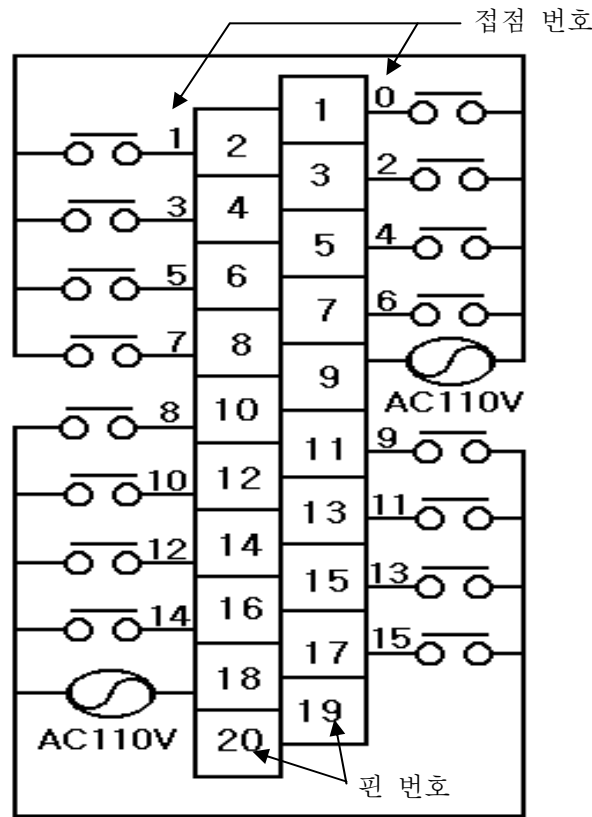


그림 1-14 G4I-A12A 외부 결선도

① TR 출력 배선(Sink 타입:G4Q-TR4A)

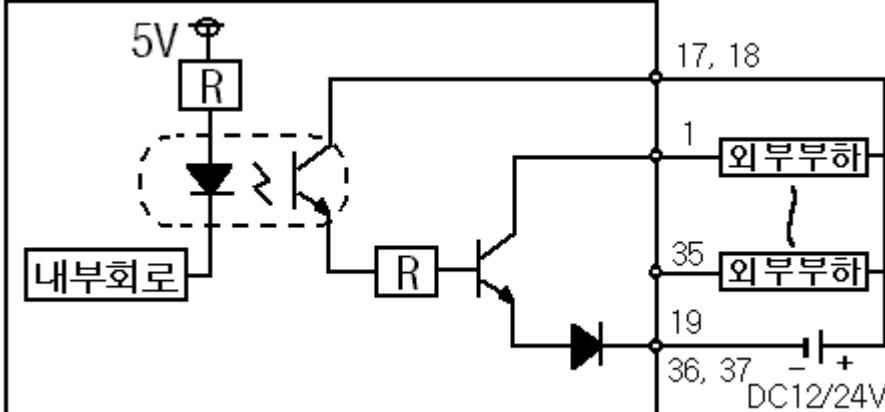


그림 1-15 TR 출력 모듈의 내부 회로(Sink 타입)

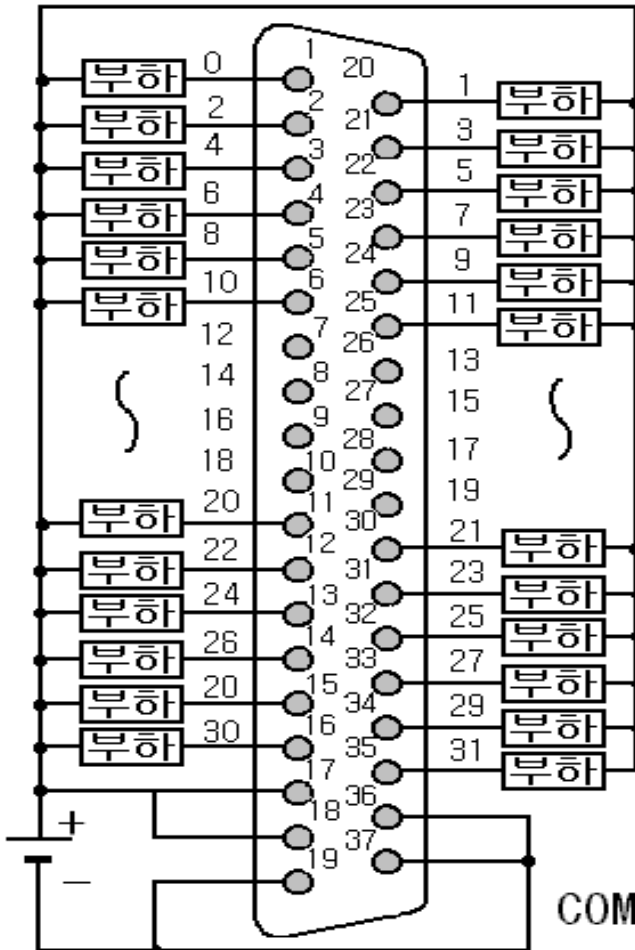


그림 1-16 G4Q-TR4A 외부 배선도(Sink 타입)

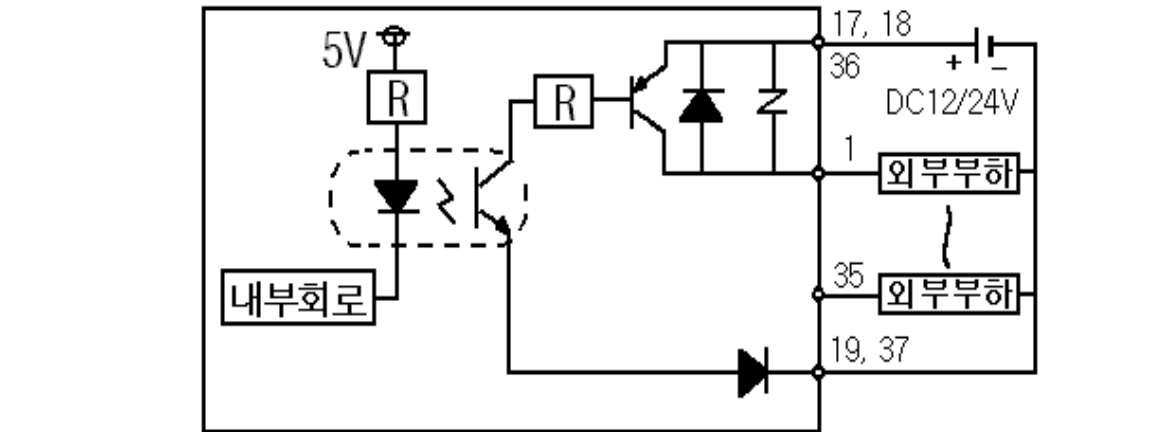


그림 1-17 TR 출력 모듈의 내부 회로(source 타입)

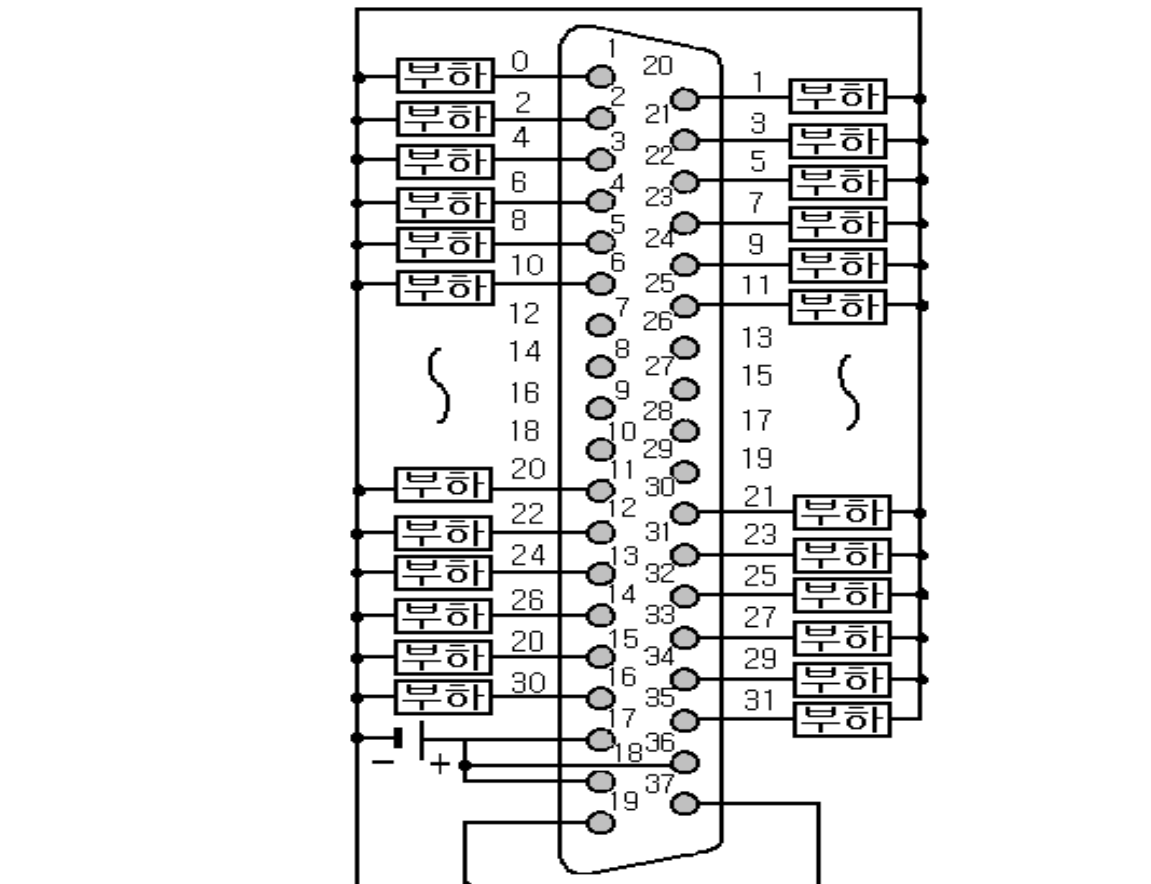


그림 1-18 G4Q-TR4B 외부 배선도(Source 타입)

③ Relay 출력 배선(G4Q-RY2A)

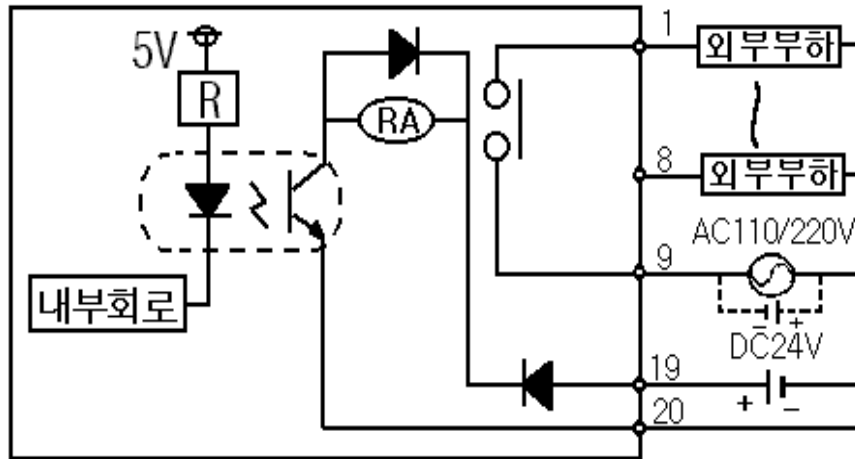


그림 1-19 릴레이 출력 모듈의 내부 회로

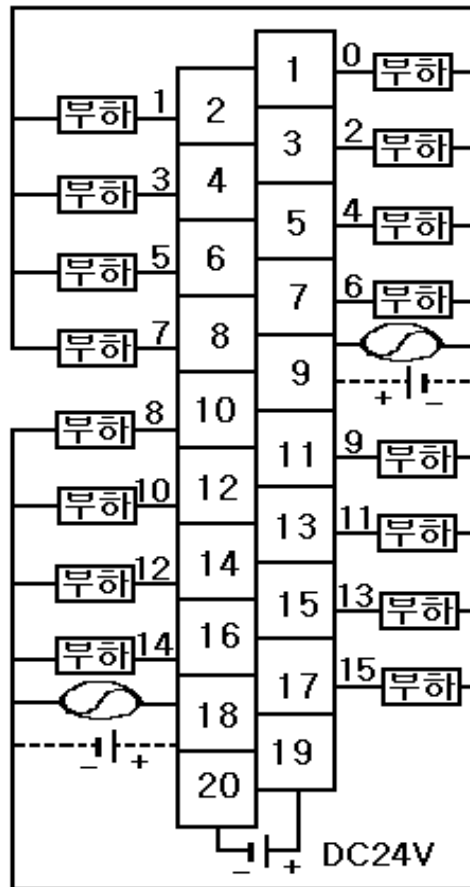


그림 1-20 G4Q-RY2A 외부 배선도(릴레이 타입)

④ SSR 출력 배선

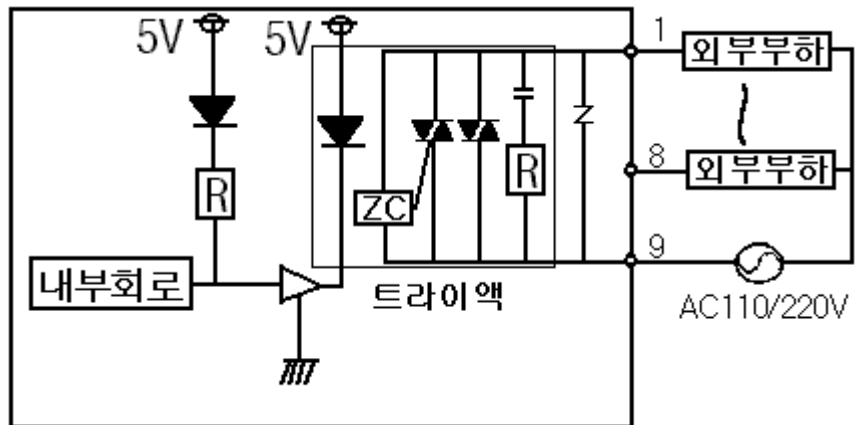


그림 1-21 SSR 출력 모듈의 내부 회로

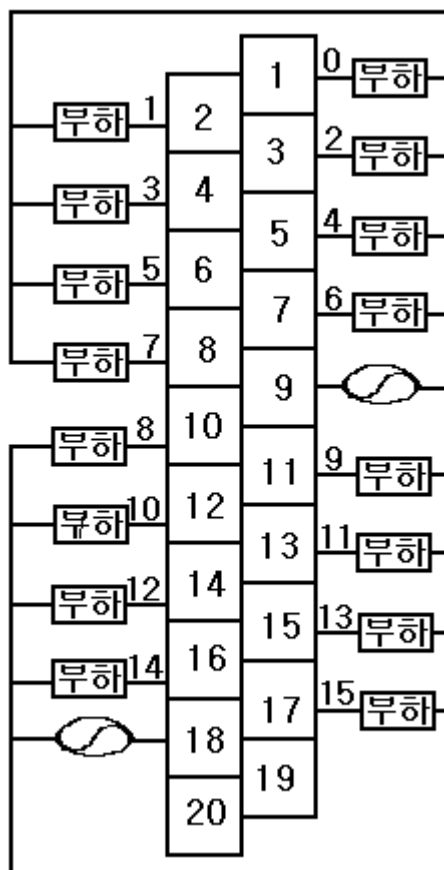


그림 1-22 G4Q-SS1A 외부 배선도(SSR 타입)

1.5 PLC의 운전 모드

GLOFA-GM의 운전 모드는 RUN, STOP, REMOTE, PAUSE로 나누어진다.

① RUN 모드(RUN)

RUN 모드는 PLC의 CPU가 정상적으로 프로그램 연산을 수행하는 모드이다.

RUN 모드는 CPU의 키를 이용해서 프로그램을 수행하는 모드로 전환하는 로컬 런 모드와 GMWIN에서 온라인 메뉴의 모드 전환에서 RUN을 시키는 REMOTE RUN이 있다. GMWIN에서 모드를 RUN으로 전환하기 위해서 CPU의 CPU의 키를 REMOTE 모드에 놓아야 한다.

② STOP 모드(STOP)

STOP 모드는 PLC의 CPU가 프로그램의 연산을 멈추고 출력을 정지시키는 모드이다.

STOP 모드는 RUN 모드와 마찬가지로 CPU의 키를 이용하여 프로그램 연산을 정지시키는 로컬 STOP 모드와 GMWIN의 온라인 메뉴의 모드 전환에서 정지시키는 REMOTE STOP이 있다. GMWIN에서 모드를 STOP으로 전환하기 위해서 CPU의 키를 REMOTE 모드에 놓아야 한다.

③ REMOTE MOTE 모드(STOP → PAU/REM)

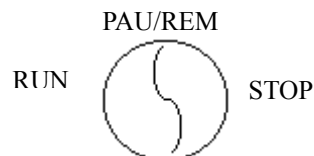
REMOTE 모드는 GMWIN에서 PLC의 모드를 변경할 수 있는 모드이다.

즉, GMWIN에서 PLC를 RUN시키거나 STOP시킬 수 있는 모드이다.

그리고 사용자 프로그램을 디버깅할 경우 REMOTE 모드에서 REMOTE STOP시킨 후 디버깅 모드로 전환이 가능하다.

④ PAUSE 모드(RUN → PAU/REM)

PAUSE 모드는 PLC의 CPU가 연산을 멈추는 모드이다. 이 때, 모든 데이터는 RUN 모드에서 PAUSE 모드로 전환되는 순간의 데이터를 유지한다.



제 2 장 GLOFA-GM 개요

그 동안 PLC 고객은 메이커(maker)마다 사용 언어와 통신 네트워크가 서로 달라 많은 불편함을 겪어 왔습니다. 이러한 불편함을 해소하고, PLC 고객에게 편리성을 도모하고자 IEC(International Electrotechnical Commission ; 국제 전기 표준 회의)에서 PLC 국제 표준화 규격이 제정 되었습니다.

국제 표준화 규격(IEC61131)은 크게 5 Part 로 구성되어 있는데

- Part 1 은 PLC 의 기본 기능 및 용어 정의
- Part 2 는 설비의 요구 기능 및 시험 조건
- Part 3 은 프로그램 언어
- Part 4 는 사용자 지침
- Part 5 는 통신 네트워크로 구성되어 있습니다.

GLOFA PLC 는 이 IEC 규격에 의해 개발되었으며 주요 특징은 다음과 같습니다.

2.1 GLOFA-GM PLC 의 특징

2.1.1 IEC 표준 언어

IEC 언어에서 새로 도입한 가장 중요한 특징들은 다음과 같습니다.

- ▷ 다양한 데이터 타입(type)을 지원합니다.
- ▷ 평선, 평선 블록, 프로그램과 같은 프로그램 구성 요소가 도입되어 상향식, 또는 하향식 설계가 가능하며 프로그램을 구조적으로 작성할 수 있습니다.
- ▷ 사용자가 작성한 프로그램을 라이브러리 화하여 다른 프로젝트에서 소프트웨어를 재 사용할 수 있습니다.
- ▷ 다양한 언어를 지원하므로 사용자는 최적의 언어를 선택하여 사용할 수 있습니다.

IEC 에서 표준화한 PLC 용 언어는 두 개의 도형 기반 언어와 두 개의 문자 기반 언어, 그리고 SFC 로 이루어져 있습니다.

(1) 도형식(graphic) 언어

- ① LD(Ladder Diagram) : 릴레이 로직 표현 방식의 언어
- ② FBD(Function Block Diagram) : 블록화한 기능을 서로 연결하여 프로그램을 표현하는 언어

(2) 문자식(text) 언어

- ① IL(Instruction List) : 어셈블리 언어 형태의 언어
- ② ST(Structured Text) : 파스칼 형식의 고 수준 언어

(3) SFC(Sequential Function Chart) : 플로우 차트(Flow Chart)와 유사한 형태로 순차적으로 전개되는 프로그램 전개 방식

현재, GLOFA PLC는 IL, LD 및 SFC 언어가 지원됩니다.

2.1.2 국제 규격의 통신 프로토콜



- ▷ Open 네트워크를 지향하여 이기종, 멀티 벤더 간의 통신이 가능합니다.
- ▷ 상위 네트워크로 Ethernet(10Mbps) 채용
- ▷ 하위 네트워크로 Fieldbus(1Mbps), Device net(500Kbps MAX.), Profibus-DP (12Mbps MAX.) 채용

2.1.3 윈도우 환경의 프로그래밍 Tool(GMWIN) 지원

- ▷ GMWIN(Programming & Debugging Tool)의 윈도우환경 채용으로 프로그램 작성, 수정 시 윈도우 장점을 모두 이용할 수 있습니다.
- ▷ MDI(Multiple Document Interface) 지원: 하나의 화면에 각기 다른 언어를 사용하여 동시에 프로그램 작성, 수정 및 모니터링이 가능합니다.

2.1.4 프로그램 작성 용이

- ▷ 프로그램의 구조화, 모듈화에 의해 프로그램 작성이 매우 편리합니다.
- ▷ 입출력 식별자명을 실제 접속되는 기기명(한글/한자 또는 영문)으로 프로그래밍이 가능합니다.

MASTER-K	GLOFA-GM
P0000 	리밋_스위치 

2.2 GLOFA-GM 성능 규격

항 목		GMR	GM1	GM2	GM3	GM4	GM6	GM7
제어 방식		저장된 프로그램 반복 연산, 정주기 연산, 인터럽트 연산						
입출력 제어 방식		스캔 동기 일괄처리 방식						
프로그램 언어		LD (Ladder Diagram) IL (Instruction List) SFC (Sequential Function Chart)						
언어 구성체 종류	오퍼레이터	LD : 13 개, IL : 21 개						
	기본 평선	156 개			194 개			
	기본 평선 블록	11 개						
	전용 평선 블록	이중화 전용 평 선 블 록	특수 기능 전용 평선 블록					
연산 처리 속도	오퍼레이터	0.12 μ s / 명령			0.2 μ s / 명령			
	기본 평선	0.12 μ s / step			0.2 μ s / 명령			
	기본 평선 블록							
프로그램 메모리용량		512 Kbyte *1			256	128	68	
최대 입출력 점수		7,680 점 *2	16,000 점	4,096 점	2,048 점	1,024 점	384 점	80 점
데이터 메모리	직접 변수 영역	0 ~ 64 Kbyte	8 ~ 64		4~32	2~16	2~8	
	네임드 변수영역 *3	256 Kbyte	446		114	52	32	
타이머		메모리 용량 내 점수 제한 없음, 시간범위 : 0.001 초~4294967.295 초(1,193 시간)						
카운터		메모리 용량 내 점수 제한 없음, 계수범위 : - 32,768 ~ + 32,767						
운전모드		Run, Stop, Pause, Debug *4						
정전시 데이터 보존		변수 정의시 보존(Retain)으로 설정된 데이터						
프로그램 블록 수		180 개					100	
프로 그램 종류	스캔	프로그램 블록 수 - 태스크 프로그램 수						
	정주기 태스크	32 개					8 개*5	
	외부접점태스크	없음	16 개			8 개	8 개*5	
	내부접점태스크	16 개					8 개*5	
	초기화 태스크	2 개(_INT, _H_INT)					1 개(_INT)	
	에러 태스크	1 개(_ERR_SYS)			없음			
자기 진단 기능		운전상태 감시, 연산지연 감시, 메모리 이상, 입출력 이상, 배터리 이상, 전원 이 상 등						
리스타트 기능		콜드, 웜, 핫 리스타트					콜드, 웜	
증설 베이스 수		15 단	31 단	7 단	3 단		불가능	3 단*6
멀티 CPU 운전		불가능	최대 4	불가능				
이중화운전		중복, 전환, 단독 입출력	불가능					

*1 : GMR-CPUB - 2 M byte (256 K step)

*2 : 네트워크 구성 시 최대 입출력 점수는 32,000 점 입니다.

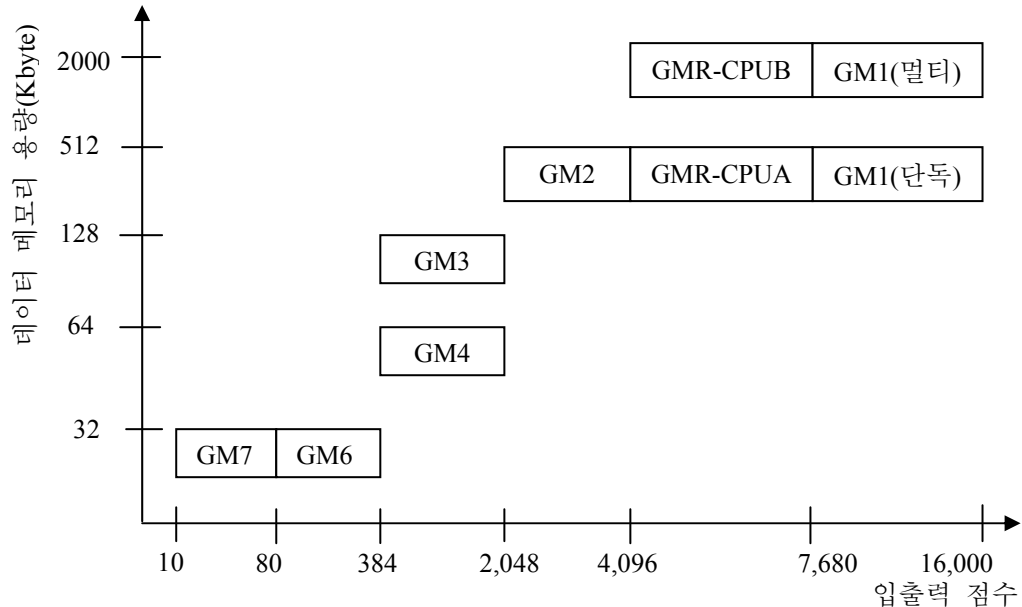
*3 : 네임드 변수 = 최대 네임드 범위 - 직접 변수 지정 범위

*4 : GMR 은 Pause 모드가 없습니다.

*5 : 태스크의 합계가 8 개 까지 가능합니다.

*6 : 증설 최대 2, 특수 최대 2, 통신 최대 1 을 조합하여 3 단까지 구성 가능합니다.

2.3 GLOFA-GM 제품 MAP



2.4 GLOFA-GM 시스템 구성

GLOFA-GM 시리즈는 각 기종별로 증설 시스템을 구성할 수 있습니다.

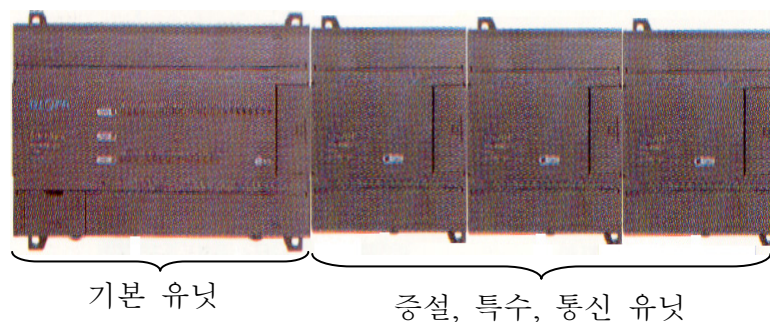
GLOFA-GMR 기종은 증설 15 단, GM1 기종은 증설 31 단, GM2 기종은 증설 7 단, GM3/4 기종은 증설 3 단까지 증설이 가능합니다.

단, GLOFA-GM4 기종의 베이스 중 12 슬롯 장착용 기본 베이스를 사용 사용할 경우 증설 시스템을 구성할 수 없으며, GM6 기종은 증설 시스템을 구성할 수 없습니다.

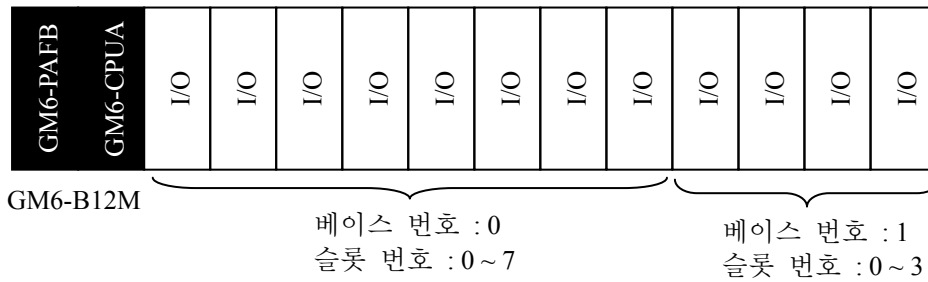
2.4.1 GLOFA-GM7 시스템 구성

GLOFA-GM7 의 기본 유닛에 전원부, 연산부, 입력부, 출력부를 모두 포함하고 있습니다. GLOFA-GM7 에는 1 개의 본체에 점점 증설 유닛 최대 2 개, 특수 유닛 최대 2 개, 통신 유닛 최대 1 개를 조합하여 3 개까지 증설할 수 있습니다.

단, 입,출력 10 점 제어용 기본 유닛에는 통신 유닛을 증설할 수 없습니다.

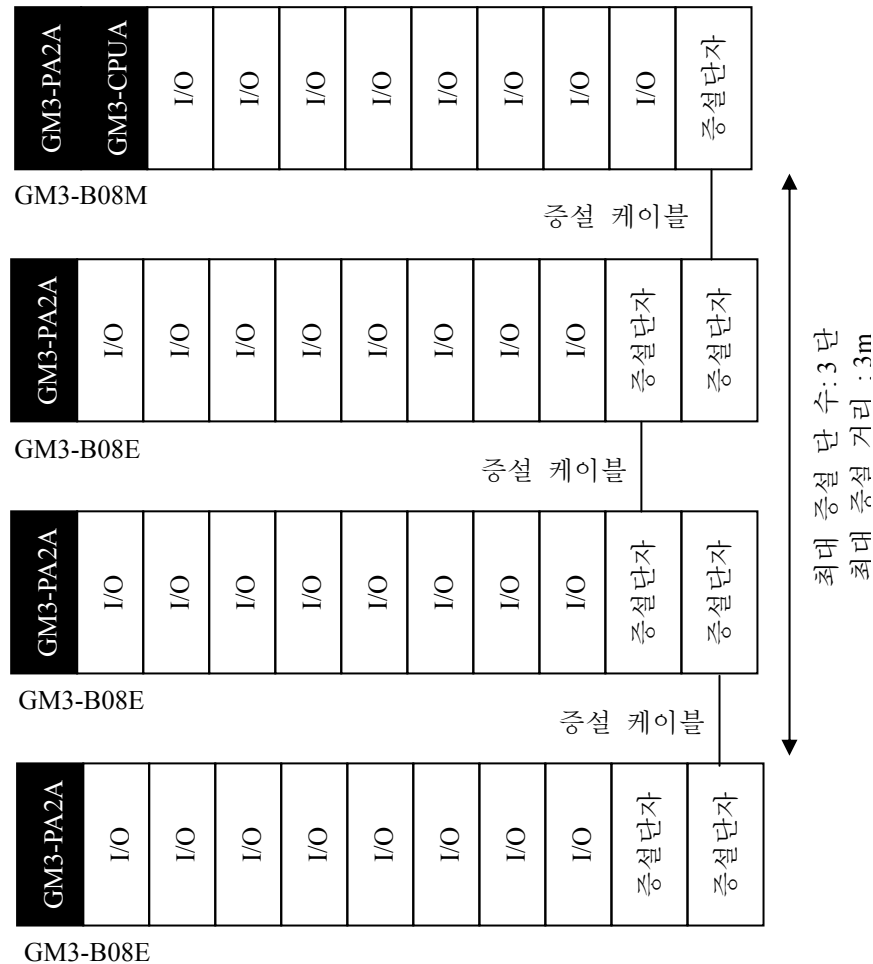


2.4.2 12 슬롯 시스템 구성(GM4/GM6)



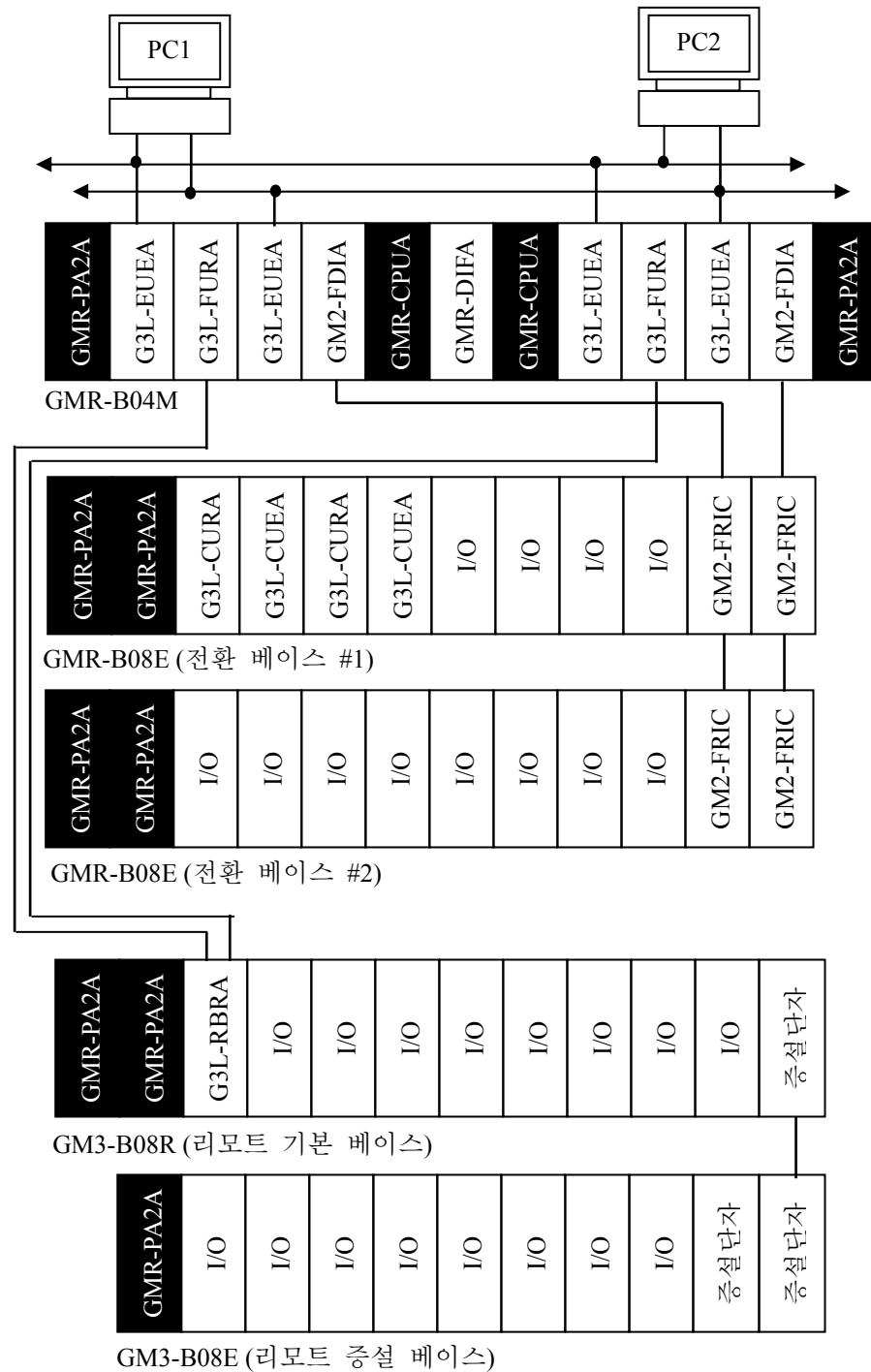
GM4 또는 GM6 에서 12 슬롯 장착용 기본 베이스(GM4-B12M/GM6-B12M)를 사용할 경우 슬롯 번호 8 번부터는 베이스 번호 1, 슬롯 번호 0 ~ 3 으로 설정하며, 통신 모듈을 장착할 수 없습니다. 그리고, GM6 에서 아날로그 입력 또는 아날로그 출력 모듈을 사용할 경우 전원 모듈은 반드시 GM6-PAFB 를 사용해야 합니다.

2.4.3 GM3/4 시스템 구성

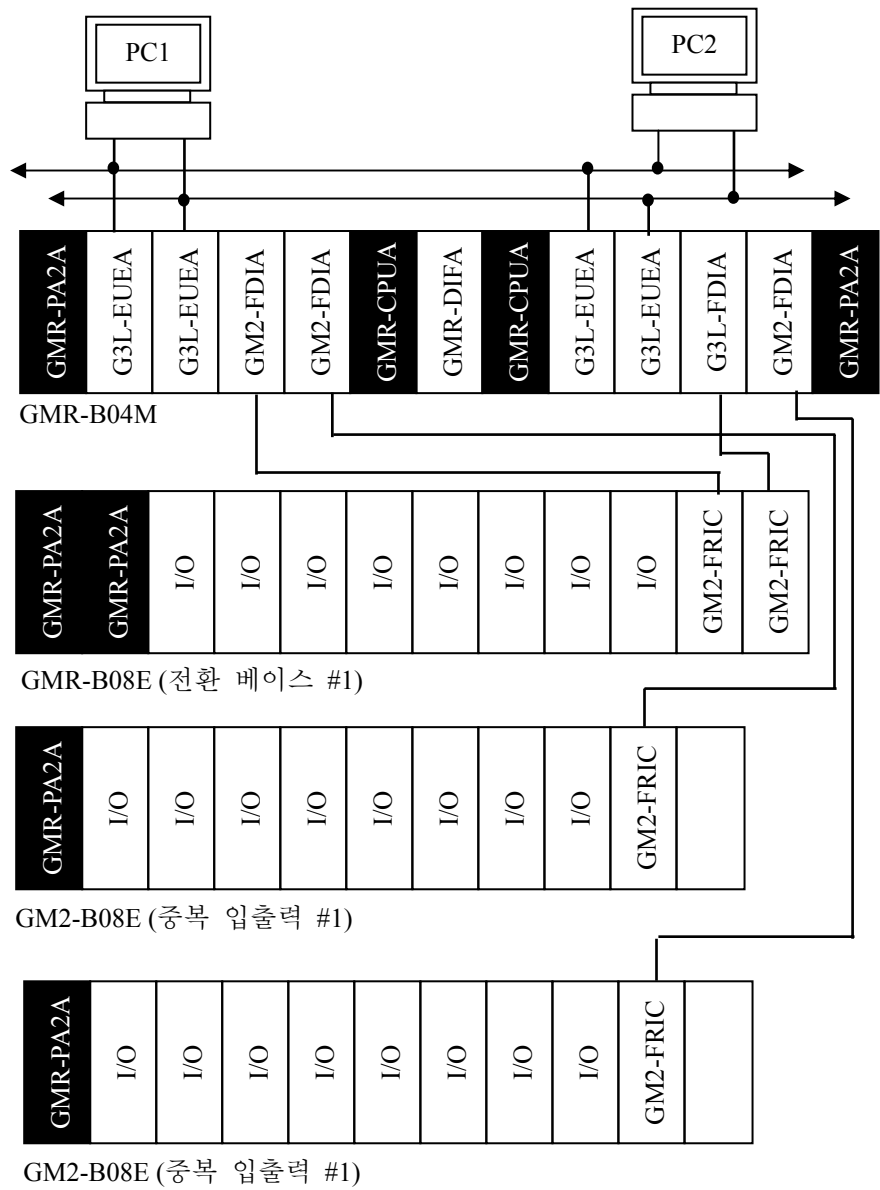


GM4 를 사용할 경우 12 슬롯 장착용 기본 베이스(GM4-B12M)를 사용할 경우, 증설 베이스를 사용할 수 없습니다.

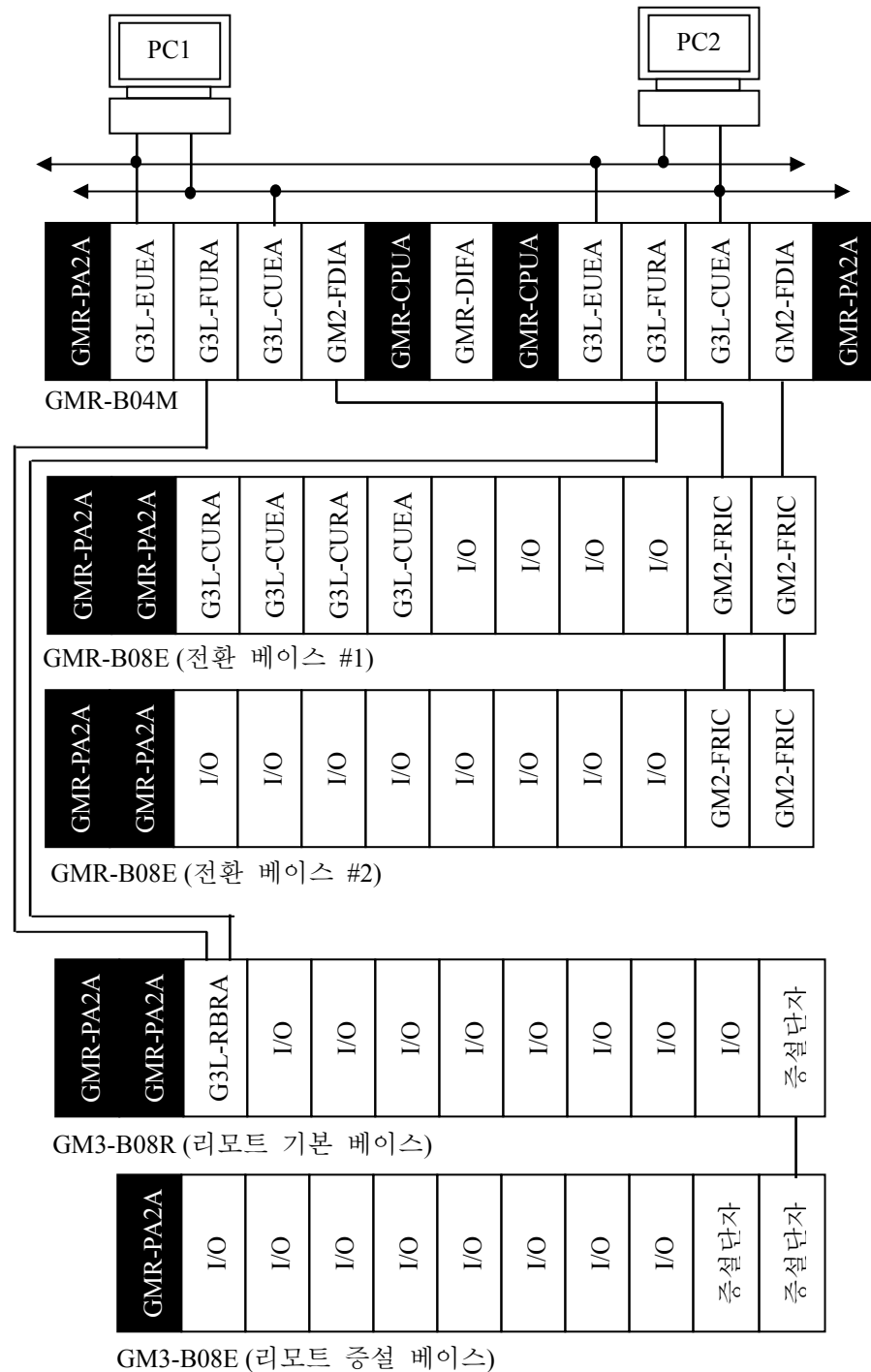
2.4.6 GMR 이중화 시스템 구성 (네트워크 이중화)



2.4.7 GMR 이중화 시스템 구성 (중복 입출력)

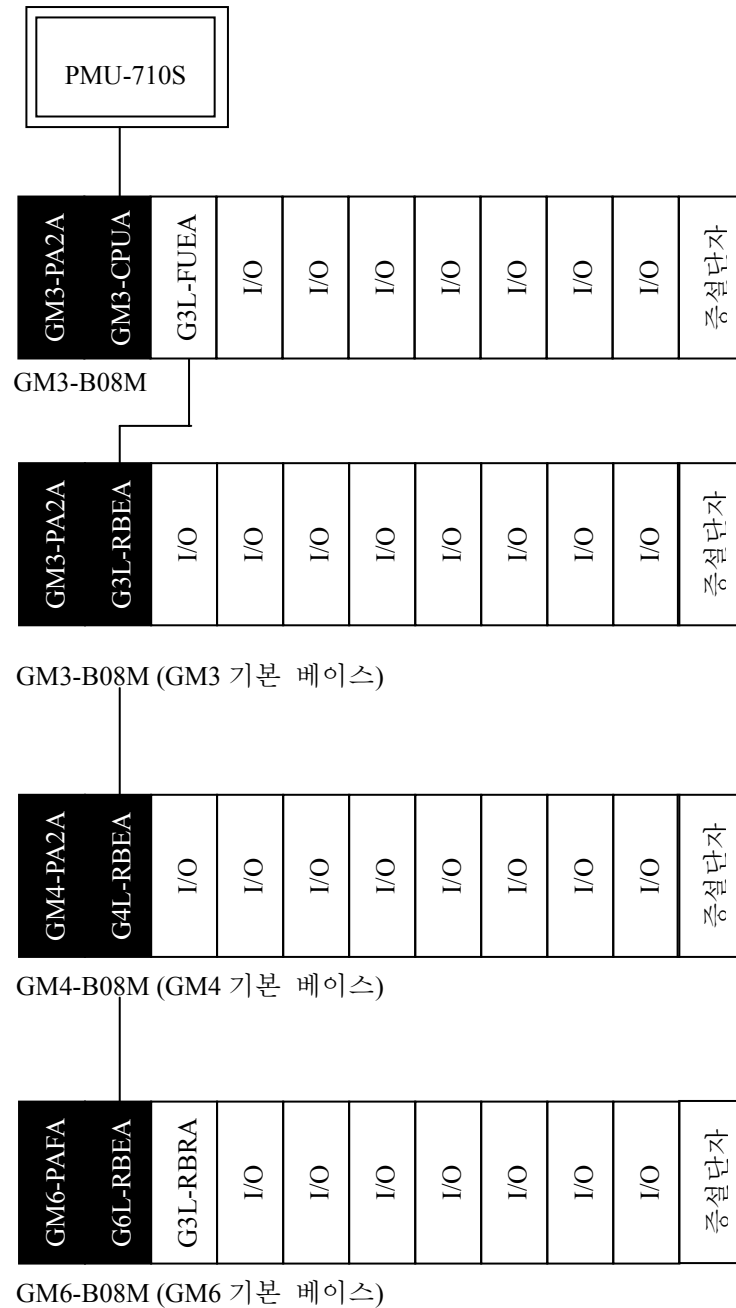


2.4.8 GMR 이중화 시스템 구성 (케이블 이중화)



알아두기 이중화 시스템에서 증설(전환 베이스, 중복 베이스, 단독 베이스)를 사용할 경우, 장거리 인터페이스 드라이버 모듈(GM2-FDIA)과 장거리 인터페이스 리시버 모듈(GM2-FRIC)을 사용해야 합니다.
(단거리 인터페이스 드라이버 모듈(GM2-NDIA) 및 단거리 인터페이스 리시버 모듈(GM2-NRIA)은 사용할 수 없습니다.)

2.4.9 GLOFA-GM 리모트 시스템 구성

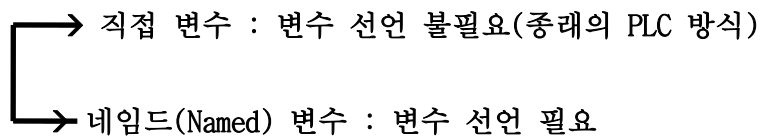


제 3 장 데이터 메모리 구성

3.1 변수의 표현 방식

프로그램 안에서 사용되는 데이터는 값을 가지고 있는데, 프로그램이 실행되는 동안에 값이 바뀌지 않는 상수와 그 값이 변하는 변수가 있습니다. 프로그램 블록, 평선, 평선블록 등의 프로그램 구성 요소에서 변수를 사용하기 위해서 우선 변수의 표현 방식을 설명 합니다.

☞ 변수 표현 방식



첫 번째 변수 표현 방식은 사용자가 이름을 부여하지 않고 이미 Maker 에 의해 지정된 메모리 영역의 식별자를 사용하는 **직접 변수** 방식이고, 두 번째 변수 표현 방식은 사용자가 이름을 부여하고 사용하는 **네임드(Named; Symbolic) 변수** 방식입니다.

3.1.1 직접 변수

직접 변수는 사용자가 변수 이름과 형 등의 선언이 없이 이미 Maker 에 의해 정해진 메모리 영역의 식별자와 주소를 사용합니다.

직접 변수에는 **%I, %Q**의 입출력 변수와 **%M**의 내부 메모리 변수가 있습니다.

입출력 변수와 내부 메모리 변수 크기는 PLC 종류에 따라 차이가 있습니다.

직접 변수는 별도의 변수 선언 없이 식별자의 위치를 표현하는 방식이므로 프로그램의 가독성(可讀性)이 떨어지며 사용자의 잘못으로 어드레스가 중복될 우려가 있습니다.

직접 변수는 반드시 퍼센트 문자(%)로 시작하고 다음에 위치 접두어와 크기 접두어를 붙이며 그리고 마침표로 분리되는 하나 이상의 부호 없는 정수의 순으로 나타냅니다.

종 류	사 용 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

3.1.1.1 입, 출력 메모리의 할당.

GLOFA-GM 시리즈 PLC의 입,출력 메모리의 할당은 다음의 5가지 인자에 의해 결정됩니다.

%IX0.0.0

① ② ③ ④ ⑤

① 위치 접두어 - 변수의 종류를 나타냅니다.

접두어	의 미
I	입력임을 나타냄.
Q	출력임을 나타냄.

② 크기 접두어 - 변수가 차지하는 메모리 공간의 크기를 나타냅니다.

접두어	의 미
X	1 비트의 크기 ("X"문자에 한하여 생략 가능)
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)의 크기

③ 베이스의 번호

CPU가 장착되어 있는 베이스(기본 베이스)를 0번 베이스라 하며, 증설 시스템을 구성했을 때 기본 베이스에 접속된 순서에 따라 베이스 번호가 증가됩니다.

GM2 이상의 기종으로 증설 시스템을 구성했을 때는 증설 베이스의 리서버 모듈에서 베이스 번호를 설정하게 되어 있으며, 설정된 베이스 번호를 사용하면 됩니다.

④ 슬롯 번호

슬롯 번호는 기본 베이스의 경우 CPU 우측이 0번이 되며 우측으로 진행하며 번호가 1씩 증가하게 됩니다.

증설 베이스의 경우 전원부 우측이 0번이 되며, 우측으로 진행하며 번호가 1씩 증가하게 됩니다.

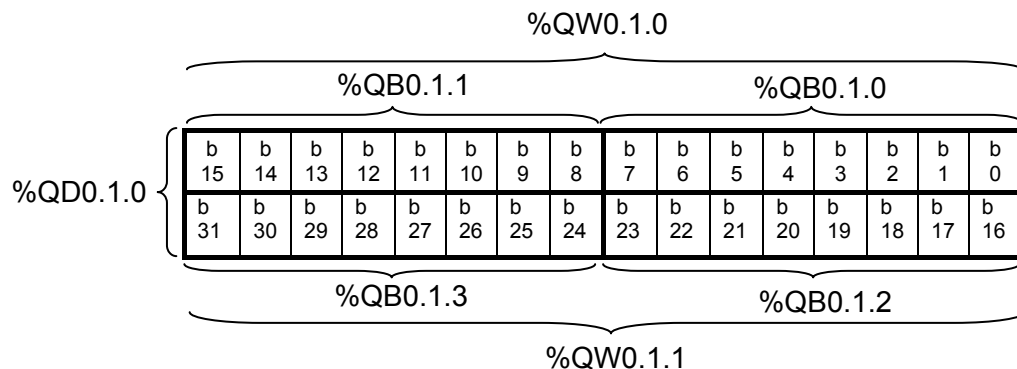
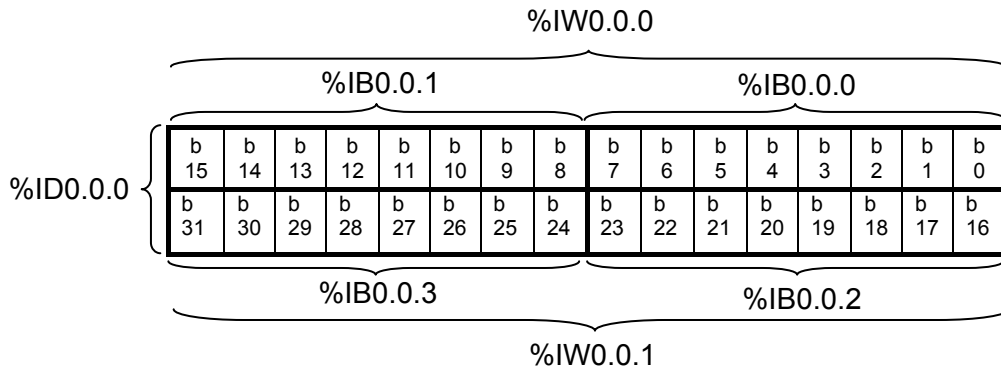
주의사항 크기 접두어 중 L(Long)은 GM4-CPUC 및 GLOFA-GM2 이상의 기종에서만 사용 가능합니다.

⑤ 크기 접두어 번호

슬롯에 장착되어 있는 접점들을 0 번 비트부터 크기 접두어 단위로 나누었을 때 몇 번째 크기 접두어 단위가 되는지를 나타냅니다.

예를 들면 0 번 슬롯에 32 점 입력 모듈이 장착되어 있고, 이것을 바이트 단위로 나누어 사용한다면 처음 8 점(%IX0.0.0~%IX0.0.7)은 %IB0.0.0 이 되고, 그 다음 8 점(%IX0.0.8~%IX0.0.15)은 %IB0.0.1 이 되며, 그 다음 8 점(%IX0.0.16~%IX0.0.23)은 %IB0.0.2 가 됩니다. 그리고 마지막 8 점(%IX0.0.24 ~%IX0.0.31)은 %IB0.0.3 이 됩니다.

그리고 1 번 슬롯에 32 점 출력 모듈이 장착되어 있고, 이것을 워드 단위로 나누어 사용한다면 처음 16 점(%QX0.1.0~%QX0.1.15)은 %QW0.1.0 이 되며, 그 다음 16 점(%QX0.1.16~%QX0.1.31)은 %QW0.1.1 이 됩니다.



주의사항 GLOFA-GM4/6 시리즈 PLC 의 베이스 중 12 슬롯 장착용 베이스 사용 시 슬롯 번호 8 번 이상은 베이스 번호 1 번, 슬롯번호 0 번부터 설정해야 합니다.

3.1.1.2 내부 메모리 할당.

내부 메모리의 할당은 위에서 설명한 입, 출력 메모리의 할당과 기본적인 방법은 동일하나 베이스 번호와 슬롯 번호를 지정하지 않습니다.

내부 메모리를 표현하는 다음의 두 가지 방법이 있습니다.

① 크기 접두어 단위의 표현

$\frac{\% \text{ M } \text{ X } \text{ N}_1}{\text{①} \quad \text{②} \quad \text{③}}$ (N_1 은 숫자)

①번 항목 %M은 내부 메모리를 나타내는 위치 접두어입니다.

②번 항목은 크기 접두어로서 입, 출력 메모리와 동일합니다.

③번 항목은 크기 접두어 번호를 나타냅니다.

② 크기 접두어를 이용한 비트 표현

$\frac{\% \text{ M } \text{ B } \text{ N}_1 \cdot \text{ N}_2}{\text{①} \quad \text{②} \quad \text{③} \quad \text{④}}$ (N_1, N_2 는 숫자)

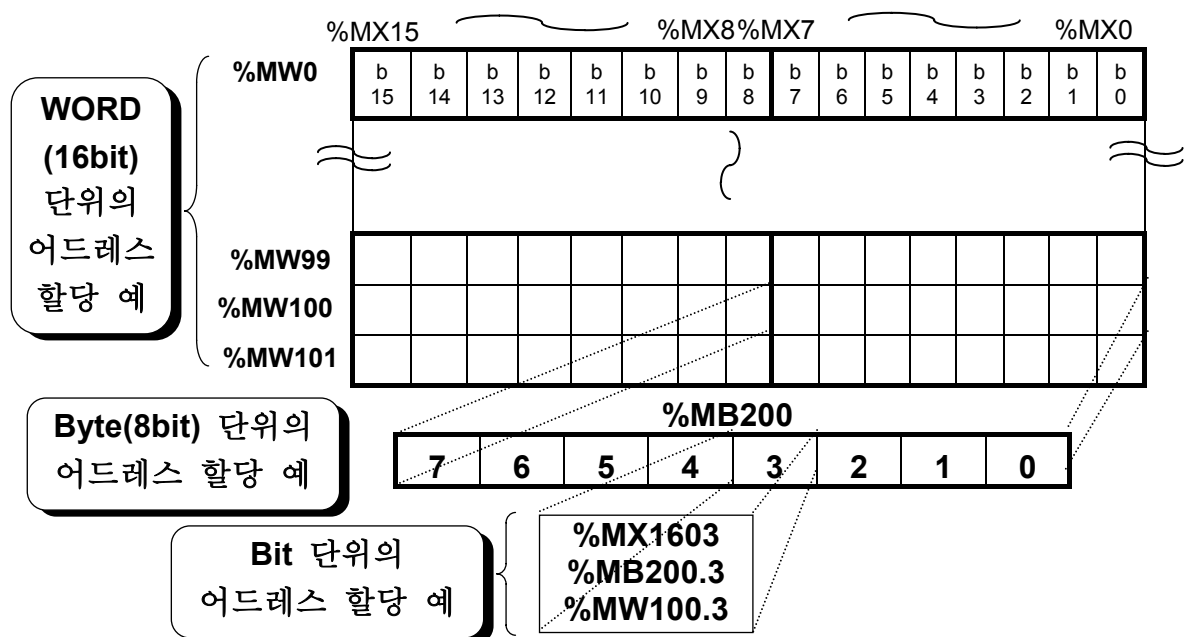
①번 항목 %M은 내부 메모리를 나타내는 위치 접두어입니다.

②번 항목은 크기 접두어로서 X를 제외한 B, W, D, L을 사용할 수 있습니다.

③번 항목은 크기 접두어 번호를 나타냅니다.

④번 항목은 비트 번호입니다.

예를 들어 %MW100.3이라고 하면 100 워드의 3번 비트를 의미합니다.



3.1.2 네임드 변수(Named Variable)

네임드 변수는 사용자가 변수 이름과 형 등을 선언하고 사용합니다.

네임드 변수의 이름은 한글/한자는 8 자, 영문은 16 자 까지 선언 가능하며 한글, 영문, 숫자 및 밑줄 문자(_)를 조합하여 사용할 수 있습니다.

또한 영문자의 경우 대·소 문자를 구별하지 않고 모두 대문자로 인식하며 빈 칸을 포함하지 않아야 합니다.

종 류	사 용 예
한글, 숫자 및 밑줄 문자	모터 10, 디지털_스위치 1, 누름_검출, 수동_배출_스위치 밸브 1, 설비_자동_운전중, 사이클_정지_완료
한글, 영문, 숫자 및 밑줄 문자	AGV_주행_완료, 모터 2_ON, BCD 값, VAL2, 자동_SOL_배출

☞ 네임드 변수의 변수 선언 절차는 다음과 같습니다.

(변수 종류 설정 → 데이터 형(type) 지정 → 메모리 할당)

① 네임드(Named) 변수의 종류(속성)

변수의 용도에 따라 다음과 같이 속성을 설정합니다.

변수 종류	내 용
VAR	읽고 쓸 수 있는 일반적인 변수
VAR_RETAIN	정전 후 복전 시 값이 유지되는 변수
VAR_CONSTANT	읽기만 할 수 있는 변수
VAR_EXTERNAL	외부 변수(VAR_GLOBAL)로 사용되는 변수

참고 사항 글로벌 변수란 하나의 프로젝트에 포함되는 여러 프로그램 블록에서 동일한 변수 이름으로 동시에 사용할 수 있는 변수입니다.

② 네임드(Named) 변수의 데이터 형(Type)

데이터의 고유 성질을 나타냅니다.

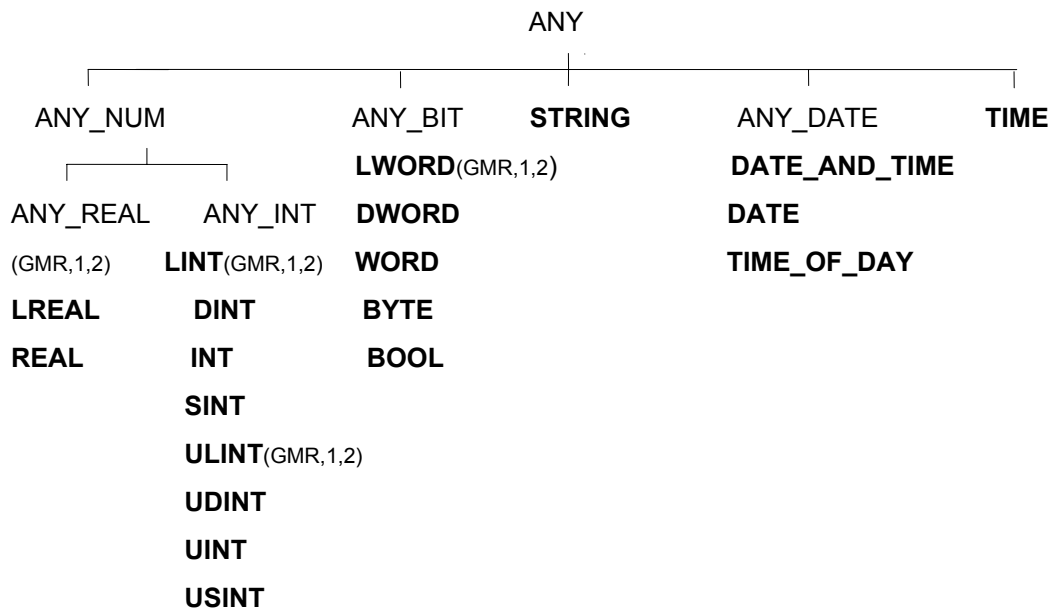
데이터형은 크게 수치(**ANY_NUM**)와 비트 상태(**ANY_BIT**)로 구분할 수 있습니다.

수치의 대표적인 경우는 정수(**INT**; Integer)인데 셀 수 있고 산술 연산을 할 수 있습니다. 정수의 예는 카운터의 현재값, A/D(아날로그 입력) 변환값 등이 있습니다.

비트 상태는 **BOOL**(1 비트), **BYTE**(8 개의 비트 열), **WORD**(16 개의 비트 열)등이 있는데 비트 열의 **On/Off** 상태를 나타내며 논리 연산을 할 수 있습니다.

비트 상태의 예는 입력 스위치의 **On/Off** 상태, 출력 램프의 소등/점등 상태 등이 있습니다. 비트 상태는 산술연산이 불가능하지만 형(**Type**) 변환 평선을 사용, 수치로 변환하면 산술 연산이 가능합니다.

BCD 는 10 진수를 4 비트의 2 진 코드로 나타낸 것이므로 비트 상태(**ANY_BIT**)입니다.



데이터 형(Type) 계층도

- ▷ **ANY_REAL**(**LREAL**, **REAL**) 및 **LINT**, **ULINT**, **LWORD** 는 **GMR**, **GM1**, **GM2** 만 사용 가능합니다.
- ▷ 앞으로 데이터 타입을 표현할 때 **ANY_NUM** 으로 나타내면 위의 계층도와 같이 **LREAL**, **REAL**, **LINT**, **DINT**, **INT**, **SINT**, **ULINT**, **UDINT**, **UINT**, **USINT** 를 모두 포함합니다.

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	-3.402823E38 ~ -1.401298E-45 1.401298E-45 ~ 3.402823E38
	LREAL	Long Reals	64	-1.7976931E308 ~ -4.9406564E-324 4.9406564E-324 ~ 1.7976931E308
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	30 문자
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

데이터 타입	초기값
SINT, INT, DINT, LINT, USINT, UINT, UDINT, ULINT	0
BOOL, BYTE, WORD, DWORD, LWORD	0
REAL, LREAL	0.0
TIME	T#0s
DATE	D#1984-01-01
TIME_OF_DAY	TOD#00:00:00
DATE_AND_TIME	DT#1984-01-01-00:00:00
STRING	"" (empty string)

데이터의 초기값

- ☞ 데이터의 초기값을 지정하지 않으면 자동적으로 위의 표와 같이 지정됩니다.
- ☞ **VAR_EXTERNAL**의 선언은 외부에서 선언한 변수를 간접 지정한 것이므로 초기값을 줄 수 없습니다.
- ☞ 변수 선언 시 **%I**와 **%Q**로 할당한 변수는 입출력에 해당하므로 초기값을 줄 수 없습니다.

③ 네임드(Named) 변수의 메모리 할당.

네임드 변수의 메모리 할당에는 **자동 할당**과 **사용자 정의**가 있습니다.

자동 할당이란 컴파일러가 내부 메모리 영역에 변수 위치를 자동으로 지정합니다. 예를 들어 “**밸브**”란 변수를 자동 메모리 할당으로 지정할 경우 변수의 내부 위치는 프로그램이 작성된 후 컴파일(Compile) 과정에서 정해지므로 사용자는 변수 위치에 신경을 쓸 필요가 없습니다. 선언된 변수는 외부 입출력과 관계 없이 내부 연산 도중 신호의 중계, 신호 상태(내부 정보)의 일시 저장, 타이머나 카운터의 점점 이름(평선 블록의 인스턴스) 지정 등에 사용됩니다.

사용자 정의란 사용자가 직접 변수(%I, %Q, %M)를 사용하여 강제로 위치를 지정합니다. 선언된 변수는 입출력용(%I, %Q) 변수와 통신 파라미터에서 사용할 통신용(%M) 변수에 사용됩니다.

사용자 정의 메모리 할당의 표현 형식은 직접변수 지정 방식과 같습니다.

3.1.3 Array(배열) 변수

Array(배열)이란 동일한 데이터형(WORD, INT, BOOL 등)으로 된 데이터가 순서대로 나열된 것을 말합니다. 이 배열을 사용하면 서로 연관된 많은 정보를 편리하게 저장할 수 있습니다.

변수를 어레이 변수로 설정을 하게 되면 데이터가 저장될 메모리 공간에 연속적으로 할당되어 데이터를 처리하는데 있어서 액세스 시간(데이터를 읽거나 쓰는데 걸리는 시간)을 줄일 수 있으므로 고속 제어를 실현할 수 있습니다.

어레이 변수의 데이터를 처리할 때는 어레이 변수 이름으로 사용하여 여러 개의 데이터를 동시에 처리 할 수 있으며, 경우에 따라서는 원소 번호를 지정함으로써 각각의 원소를 개별적으로 처리할 수도 있습니다.

원소번호	내용	데이터 형	원소 번호	내용	데이터 형
_RTC_TIME[0]	년도	Byte	_RTC_TIME[4]	분	Byte
_RTC_TIME[1]	월	Byte	_RTC_TIME[5]	초	Byte
_RTC_TIME[2]	일	Byte	_RTC_TIME[6]	요일	Byte
_RTC_TIME[3]	시	Byte	_RTC_TIME[7]	백년	Byte


제 4 장 프로그램 편집 TOOL(GMWIN)

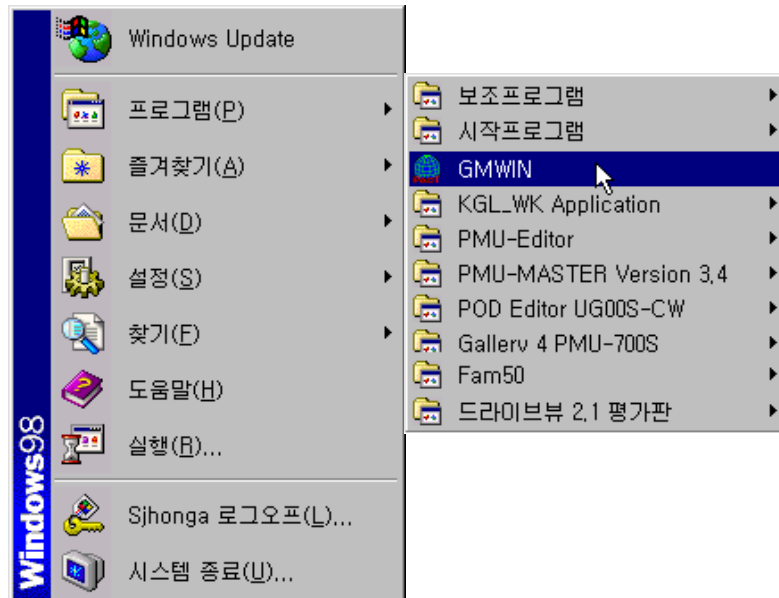
4.1 기본 사용법

GMWIN 은 GLOFA-GM PLC 의 프로그램을 편집하고 실행 파일을 만들어 PLC 에 전송하며 PLC 의 데이터를 모니터링, 디버깅하는 소프트웨어 툴입니다.

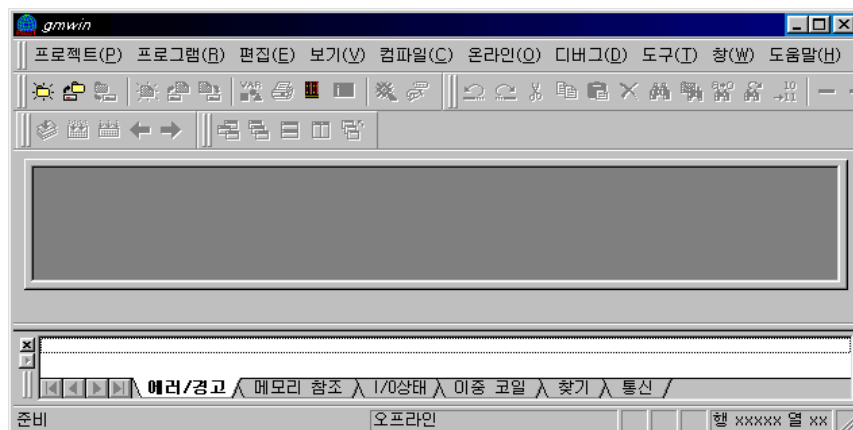
GMWIN 은 다중 문서 인터페이스(MDI:Multiple Document Interface)방식으로 동시에 여러 개의 프로그램을 편집, 모니터링할 수 있습니다.


4.1.1 프로젝트 및 프로그램 정의

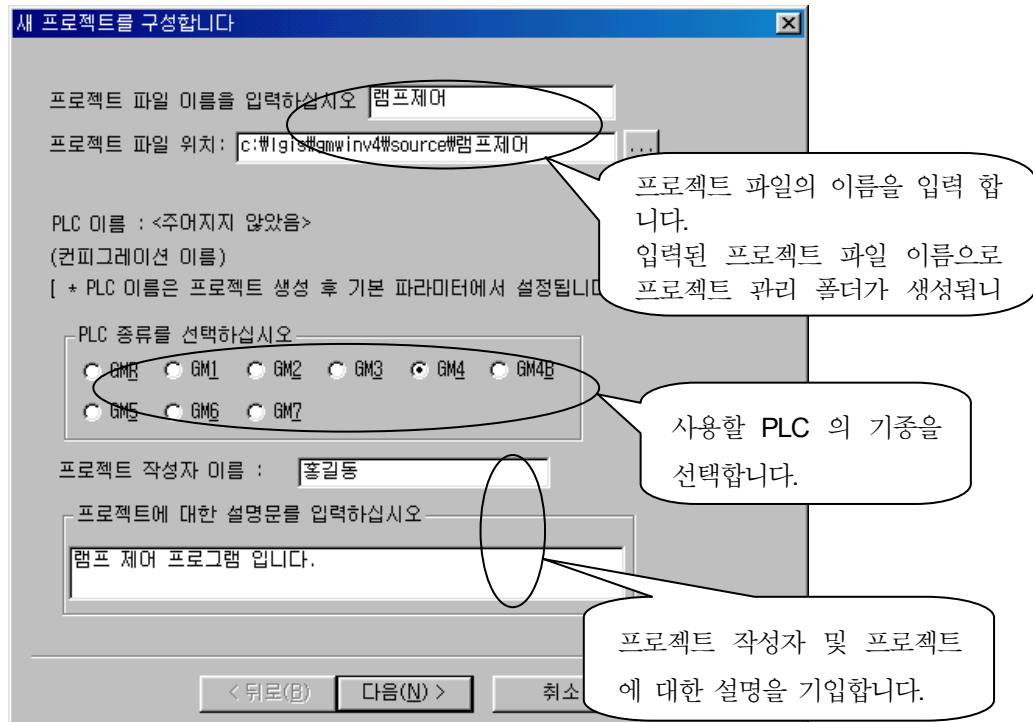
- ◆ 윈도우의 시작 메뉴( 시작)를 누르고 프로그램 > GMWIN 을 선택하거나 바탕 화면의 GMWIN 단축 아이콘을 클릭합니다.



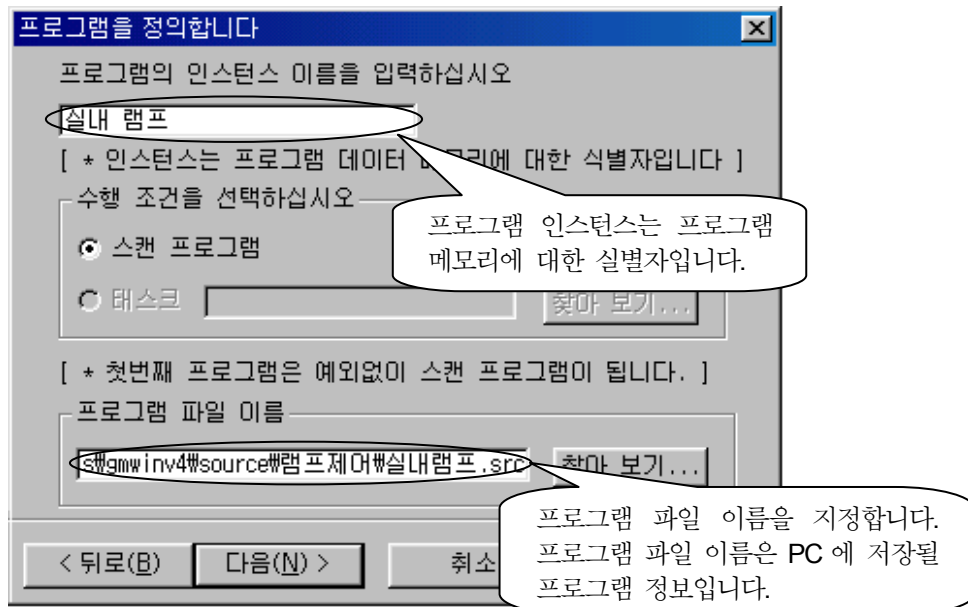
- ◆ 아래와 같은 GMWIN 초기화면이 나옵니다.



- ◆ 프로젝트(P) 메뉴 >> 새 프로젝트(N)를 클릭하거나 새 프로젝트 단축 아이콘 ()을 클릭하여 새 프로젝트 대화 상자를 부릅니다.



- ◆ 생성된 새 프로젝트 화면 입력란에 입력을 완료한 후 다음(N)을 클릭하여 프로그램을 정의합니다.

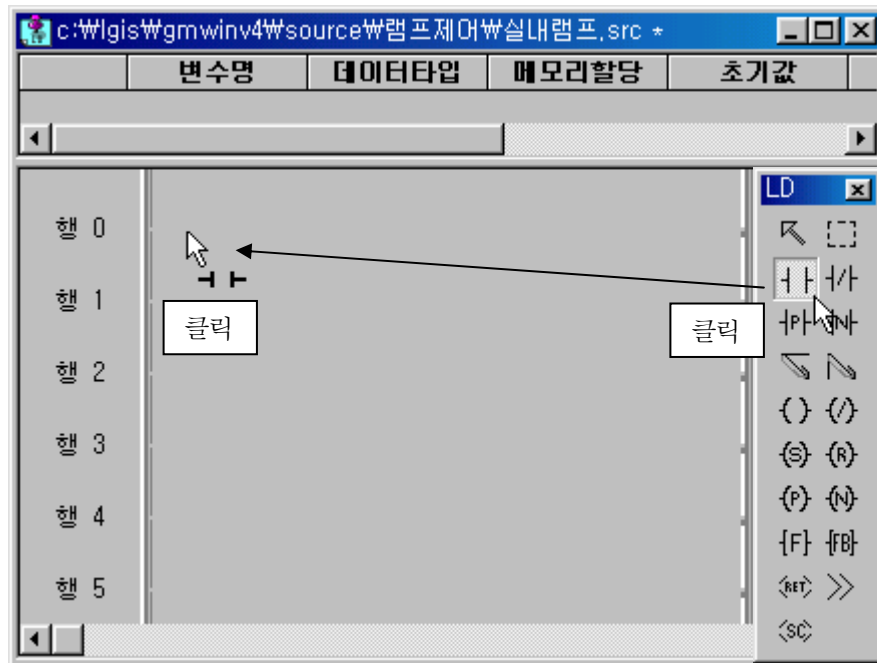


알아두기 스캔 프로그램과 태스크

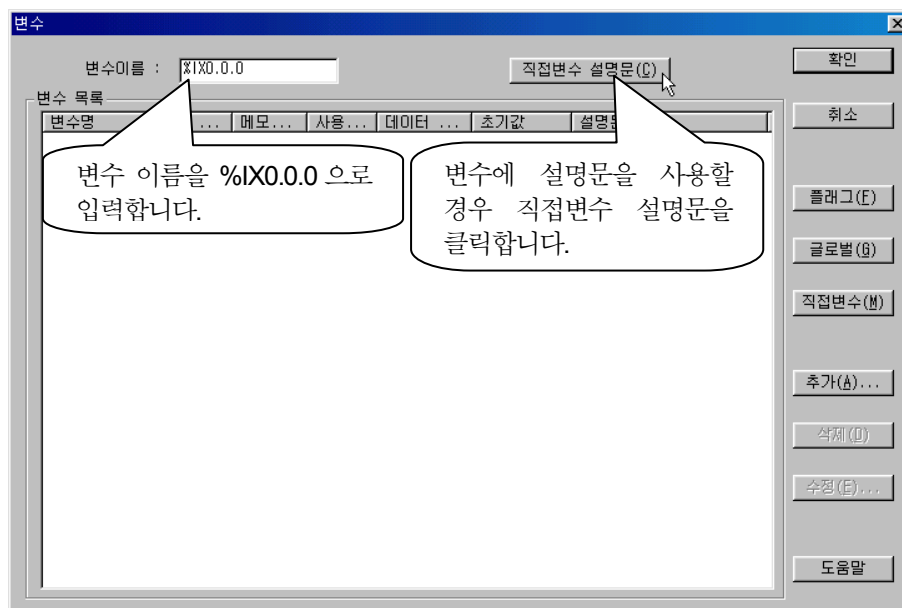
GLOFA-GM에서 실행되는 프로그램은 스캔 프로그램과 태스크 두 종류가 있습니다. 스캔 프로그램은 CPU 가 RUN 상태이면 조건 없이 실행되는 프로그램이고, 태스크는 CPU 가 RUN 상태이면서 특정 조건을 만족해야만 수행되는 프로그램입니다.

4.1.2 프로그램의 편집

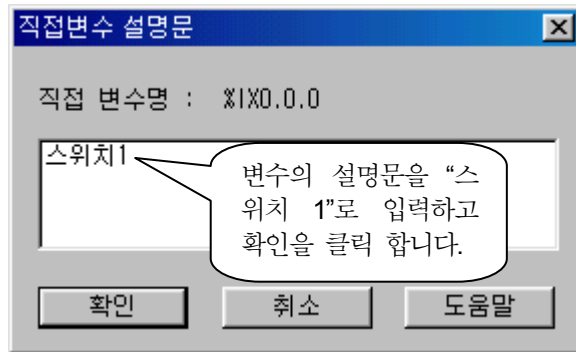
- ◆ 프로그램에 평상시 열린 접점(a 접점)을 편집하기 위해서는 도구상자에서 평상시 열린 접점을 클릭하여 프로그램 창에서 다시 한 번 클릭합니다.



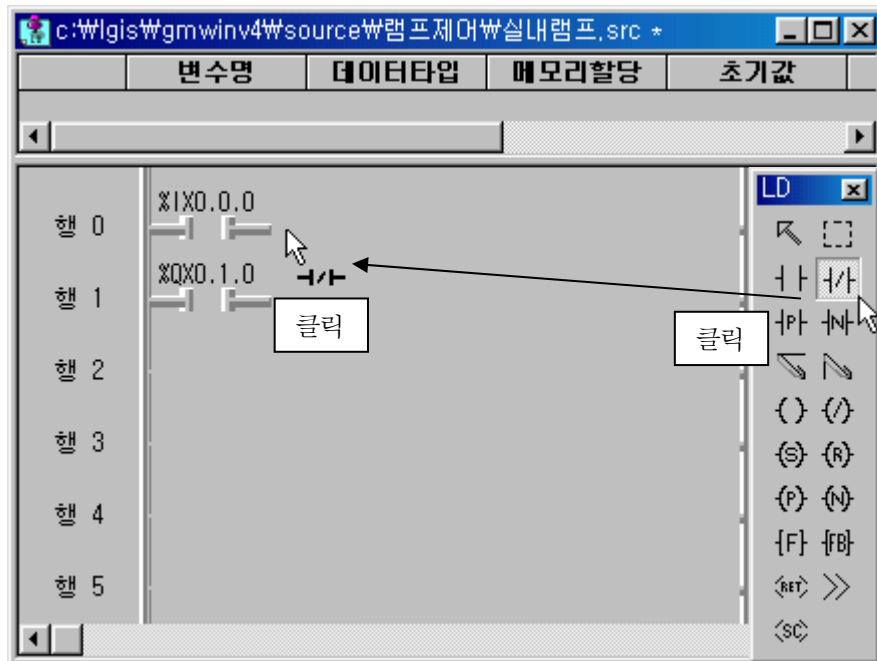
- ◆ 프로그램 창에서 클릭을 하면 변수 입력 창이 나타납니다.
변수 이름을 입력하고 변수 설명문을 입력하고자 할 경우 직접 변수 설명문을 클릭합니다.



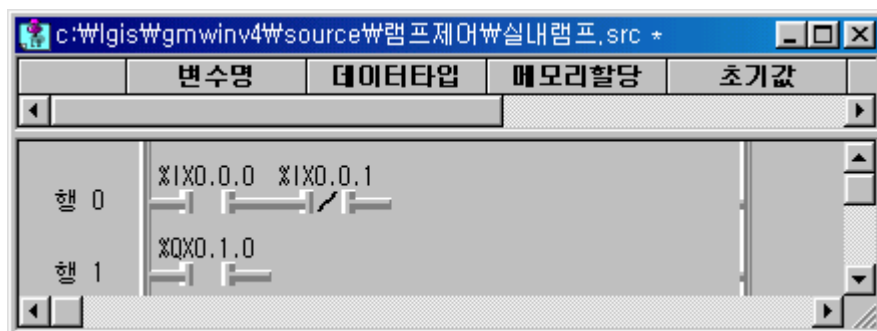
- ◆ 직접변수 설명문을 클릭하면 그림과 같은 직접변수 설명문 입력 창이 나타납니다.



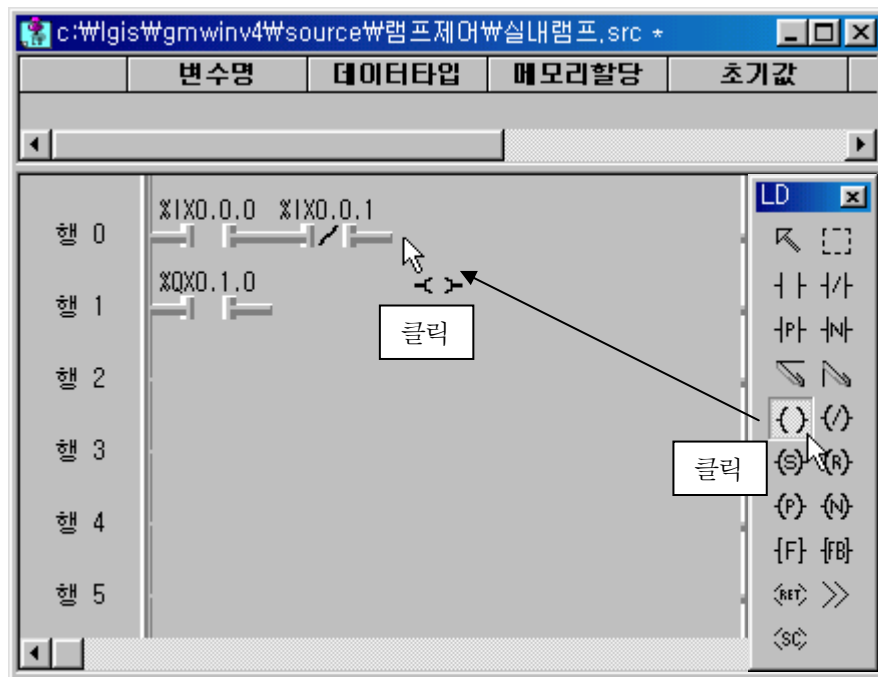
- ◆ 프로그램에 평상시 닫힌 접점(b 접점)을 편집하기 위해서는 도구상자에서 평상시 닫힌 접점을 클릭하여 프로그램 창에서 다시 한 번 클릭합니다.



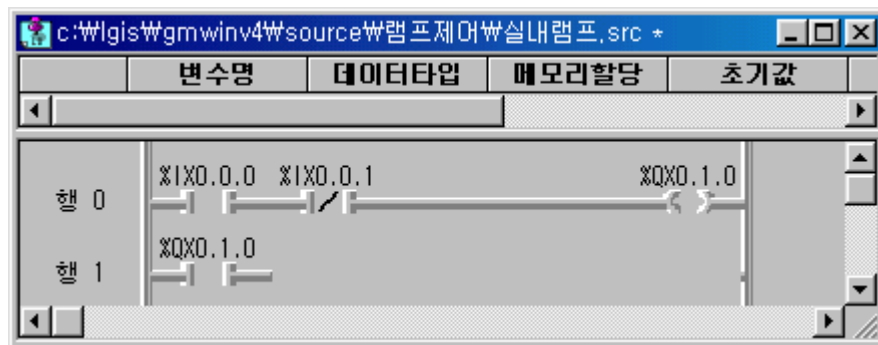
- ◆ 도구 상자에서 평상시 닫힌 접점(b 접점)을 선택한 후 프로그램 창에서 클릭하면 평상시 열린 접점의 변수 입력과 동일한 변수 입력 창이 나타납니다. 변수 이름에 “%IX0.0.1”로 입력한 후 직접변수 설명문을 클릭하여 “스위치 2”라는 직접변수 설명문을 입력합니다.



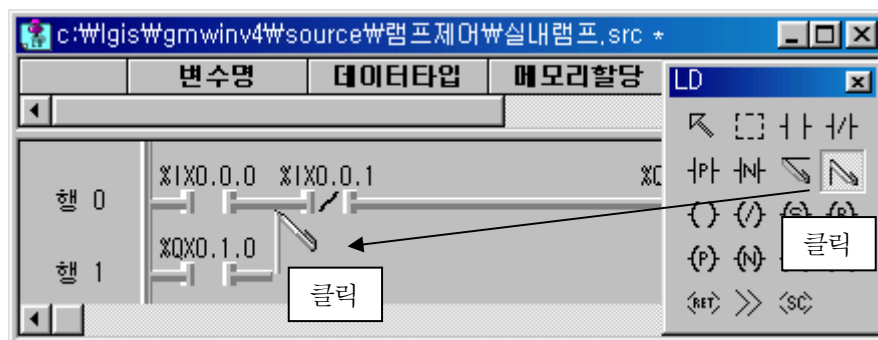
- ◆ 프로그램에 출력 코일을 편집하기 위해서는 도구상자에서 출력 코일을 클릭하여 프로그램 창에서 다시 한 번 클릭합니다.



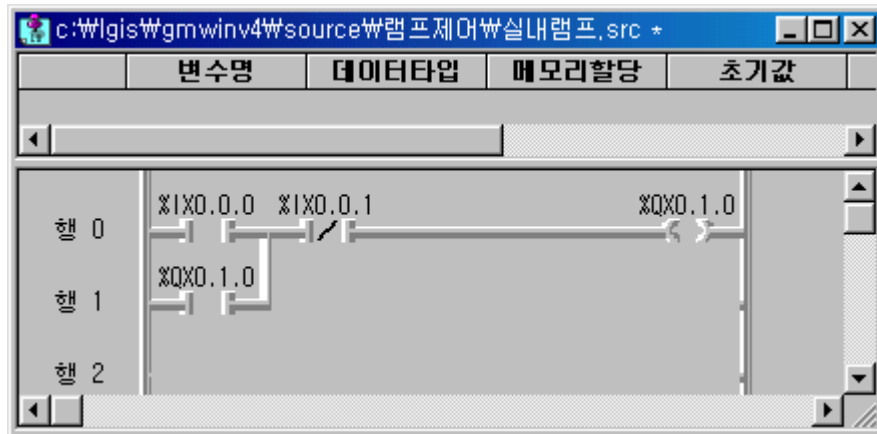
- ◆ 도구 상자에서 코일을 선택한 후 프로그램 창에서 클릭하면 평상시 열린 접점의 변수 입력과 동일한 변수 입력 창이 나타납니다. 변수 이름에 "QIX0.1.0"으로 입력한 후 직접변수 설명문을 클릭하여 "램프"라는 직접변수 설명문을 입력합니다.



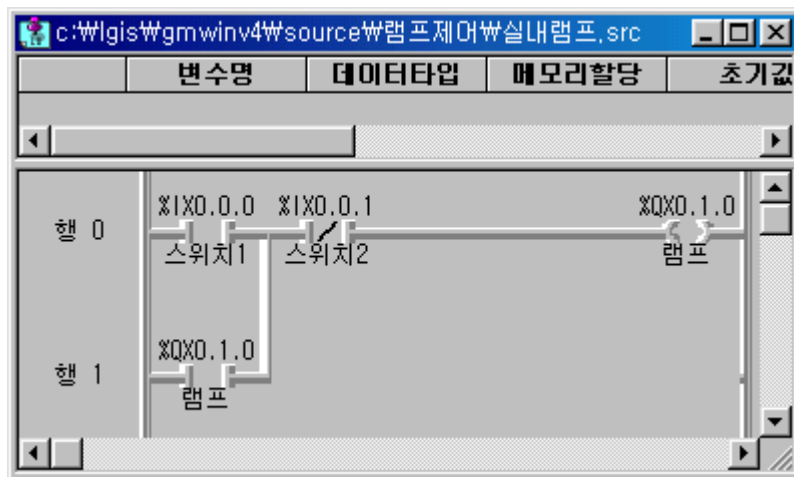
- ◆ 수직으로 분리된 두 회로를 연결하기 위하여 수직선을 사용합니다.



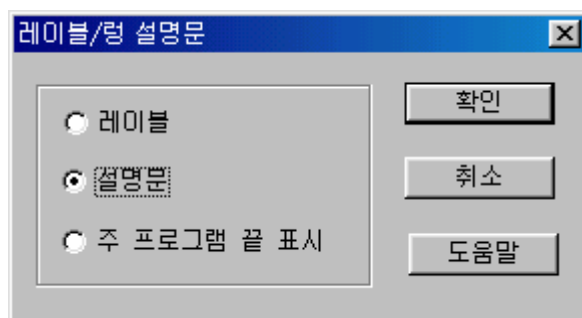
- ◆ 수직선을 연결하여 두 회로를 연결하면 프로그램 작성이 완료됩니다.



- ◆ 보기 메뉴 >> 메모리 위치/설명문을 선택하면 작성한 프로그램에서 각 변수에 설정된 설명문을 볼 수 있습니다.

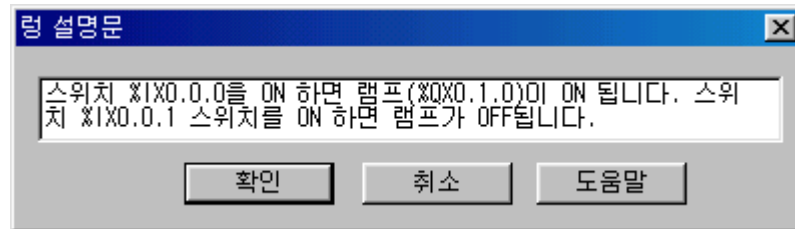


- ◆ 프로그램의 부분 동작에 관한 설명문을 프로그램에 삽입할 수 있습니다.
설명문을 삽입하고자 하는 행 번호를 더블 클릭하면 다음과 같은 레이블/링 설명문 화면이 나타납니다.



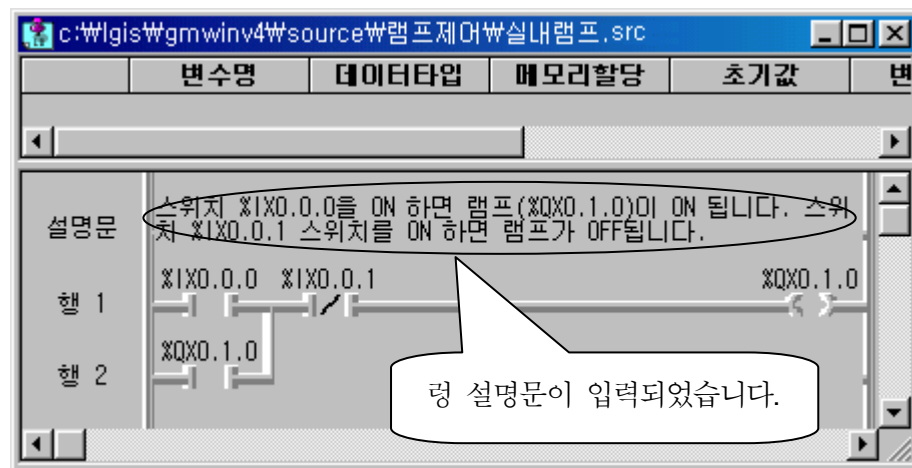
- ◆ 설명문을 선택한 후 “ 확인” 을 클릭하면 다음과 같은 령 설명문 화면이 나타납니다.

령 설명문을 입력하고 확인을 클릭합니다.



- ◆ 다음 그림과 같이 설명문이 입력되었습니다.

프로그램에서 연결된 입력 또는 출력의 단위를 령이라고 하며, 령에 관한 설명문을 령 설명문이라고 합니다.



4.1.3 컴파일 및 메이크

GLOFA-GM PLC 는 PC 에서 작성한 프로젝트 소스를 그대로 인식할 수 없습니다. CPU 모듈에 들어있는 마이크로 프로세서는 0 과 1 로 구성되는 기계어만을 인식할 수 있기 때문입니다.

PC 에서 작성한 소스 파일을 기계어로 바꾸어주는 과정을 ‘컴파일’이라고 하며 소스 파일 이외의 프로젝트 내 항목들을 연결시켜 주는 과정을 ‘메이크’ 라고 합니다.

컴파일 메뉴의 ‘모두 컴파일’을 실행하면 컴파일과 메이크가 동시에 실행됩니다.

사용자가 프로젝트를 작성하고 프로그램을 편집하여 컴파일 및 메이크를 실행하면 다음과 같은 파일이 만들어 집니다.

<프로젝트 명>.PRJ : 사용자가 작성한 프로젝트 파일

<프로젝트 명>.SRC : 사용자가 작성한 프로그램 파일

<묶음 파일>.MUK : 프로젝트에 대한 묶음 파일

<프로젝트 명>.BNO : PLC 실행 파일.

GM1 인 경우, 리소스 개수만큼 생깁니다.

<프로젝트 명>.BNO ~ <프로젝트 명>.BNi, (i 는 리소스 개수)

<프로젝트 명>.MON : 모니터링을 위한 정보 파일

<프로젝트 명>.CRO : 메모리 참조 실행 파일을 만들 때 생성. 각각의 프로그램에서 사용한 글로벌 변수 및 직접 변수를 나타낸 텍스트 파일 (Cross Reference)

<프로젝트 명>.DLO : upload 파일

<프로젝트 명>.EW0 : 런 중 수정을 위한 파일

<프로젝트 명>.INF : 모니터 및 디버그를 위한 파일

<프로젝트 명>.TWO : 런 중 수정을 위한 파일

<프로젝트 명>.VAR : 변수모니터에서 사용자가 지정한 변수들을 보관한 파일

<프로그램 명>.SRC : 사용자가 작성한 프로그램 파일

<프로그램 명>.ASV : 사용자가 작성한 프로그램을 이 이름으로 주기적으로 저장합니다. 메뉴 Option-Auto 저장에서 타임 값을 설정 하였을 경우에만 생성되고 정상적으로 프로그램 창을 닫은 경우에는 이 파일은 자동으로 삭제됩니다.

<프로그램 명>.MPS : 시뮬레이터의 데이터 램 파일

<프로그램 명>.OP? : 프로그램을 컴파일 하면 생성됩니다.(프로그램 블록인 경우)

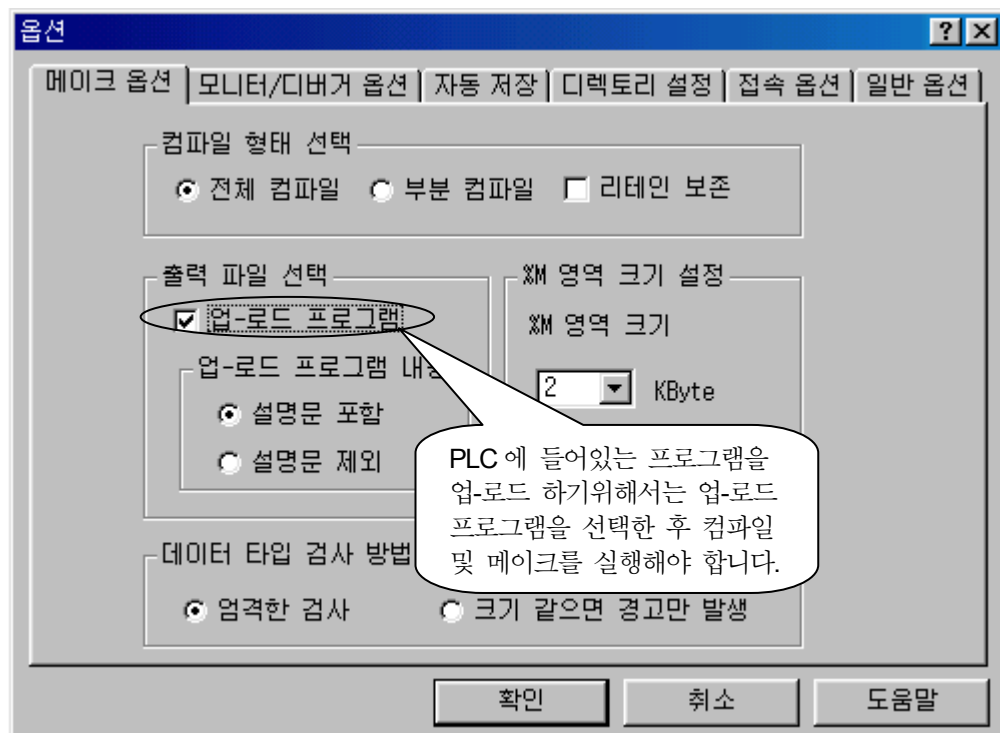
<프로그램 명>.OB? : 프로그램을 컴파일 하면 생성됩니다.(평선 블록인 경우)

<프로그램 명>.OF? : 프로그램을 컴파일 하면 생성됩니다.(평선인 경우)

(OP3 : GM3 인 경우, OP4 : GM4 인 경우)

- <프로그램 명>.PCI : 부분 컴파일 정보 파일
 <프로그램 명>.PCB : PCI Backup 파일
 <프로그램 명>.SP? : 프로그램을 시뮬레이션 컴파일 하면 생성됩니다.
 (프로그램 블록인 경우)
 <프로그램 명>.SB? : 프로그램을 시뮬레이션 컴파일 하면 생성됩니다.
 (평선 블록인 경우)
 <프로그램 명>.SF? : 프로그램을 시뮬레이션 컴파일 하면 생성됩니다.
 (평선인 경우)

프로그램을 PLC 로 전송한 후 PLC 에 들어있는 프로그램을 다시 PC 로 불러내기(업-로드) 위해서는 컴파일 및 메이크를 실행할 때 업-로드 정보를 포함해서 컴파일 및 메이크를 실행해야 합니다. 업-로드 정보를 포함해서 컴파일 및 메이크를 하기 위해서는 **프로젝트 메뉴 >> 옵션의 메이크 옵션 탭**에서 그림과 같이 출력 파일에서 업-로드 프로그램을 선택한 후 컴파일 및 메이크를 실행해야 합니다.



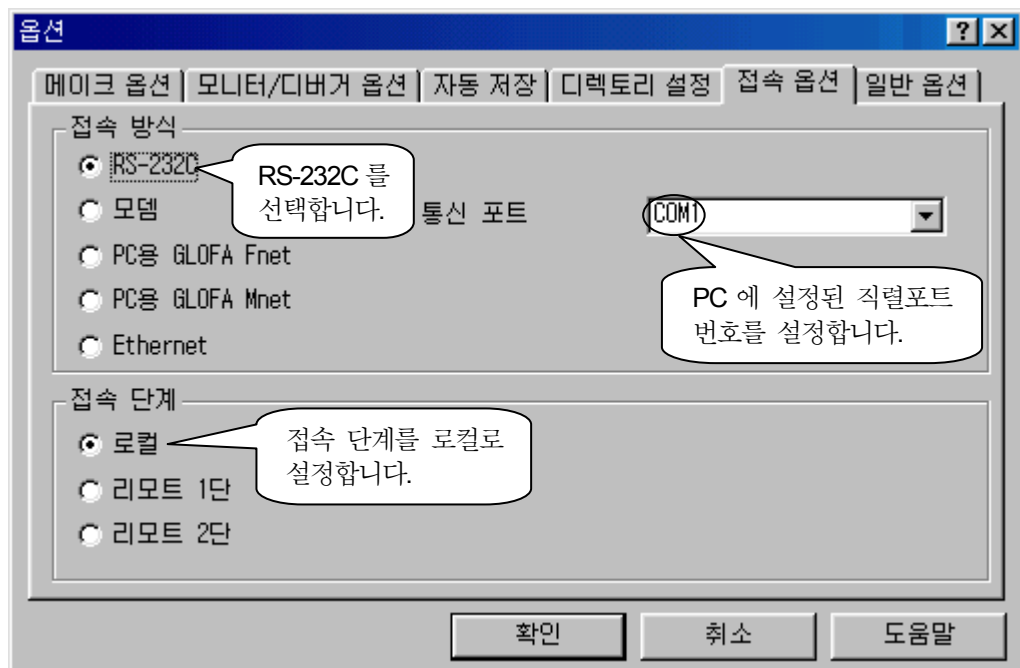
알아두기 밀줄 친 프로젝트 파일과 프로그램 파일은 반드시 보관하여야 할 파일이며 나머지 파일들은 컴파일 및 메이크 실행으로 다시 생성 할 수 있습니다.


4.1.4 접속 및 전송

4.1.4.1 RS-232C 을 이용한 접속

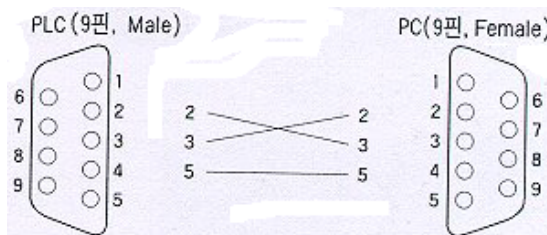
PC 에서 작성한 프로젝트를 PLC 로 전송하기 위해서는 PC 와 PLC 사이에 통신이 연결되어야 합니다. RS-232C 를 이용한 접속 방식은 GMWIN 에서 지정한 PC 의 직렬 포트와 PLC CPU 포트를 RS-232C 케이블로 연결하여 PC 에서 작성한 프로젝트 정보를 PLC 로 전송합니다.

- ◆ 프로젝트 메뉴 >> 옵션을 선택하고 접속 옵션 탭을 선택하여 다음과 같이 설정합니다.

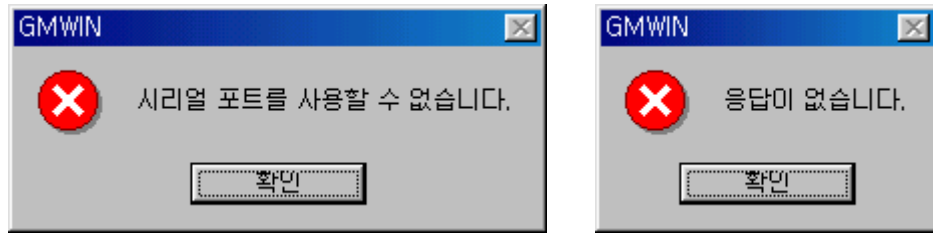


- ◆ 접속 옵션을 설정한 후 온라인 메뉴 >> 접속을 클릭하거나 단축 아이콘에서 접속 아이콘()을 클릭하여 접속이 이루어지면 PC 에서 작성한 프로젝트를 PLC 로 전송할 준비가 완료됩니다.

알아두기 PC 와 PLC 연결 케이블의 결선은 다음과 같습니다.



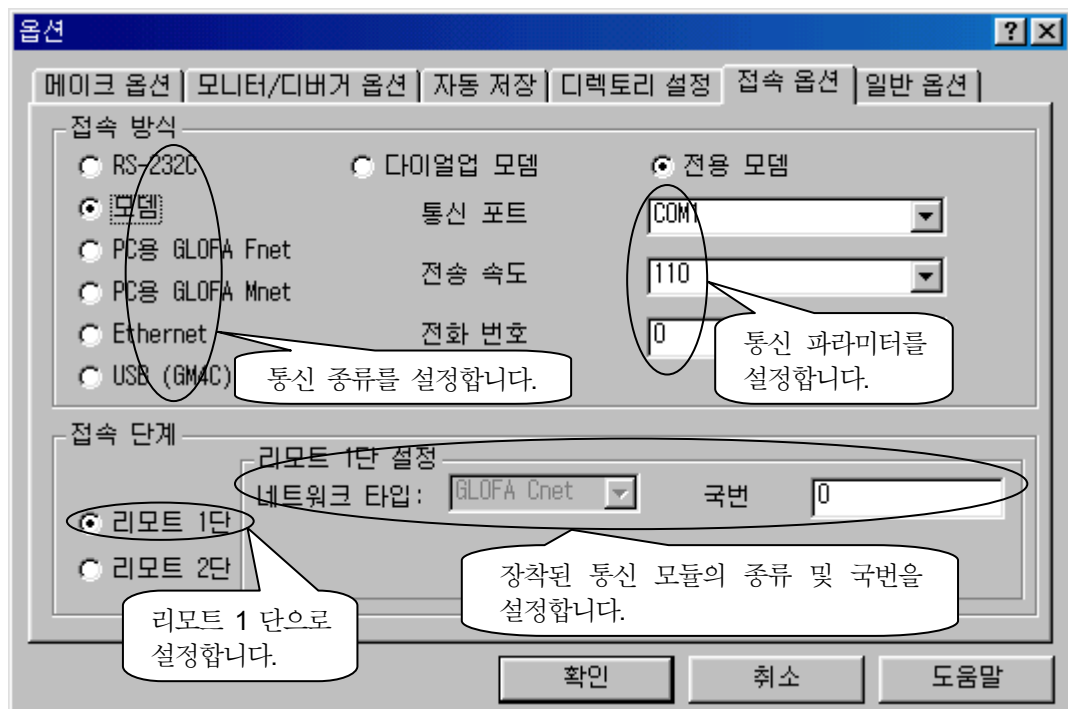
- ◆ 온라인 메뉴에서 접속을 클릭하거나 접속 아이콘을 클릭했을 때 다음과 같은 메시지가 나오면 접속 또는 연결 케이블을 확인해 주십시오.



4.1.4.2 통신 모듈을 이용한 리모트 접속

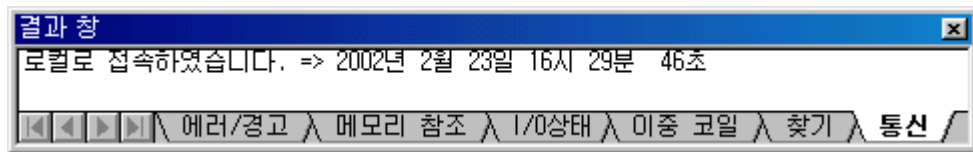
PC 에서 작성한 프로젝트를 CPU 로 전송할 때 CPU 에 있는 통신 포트뿐만 아니라 CPU 에 통신 모듈이 장착되어 있다면 통신 모듈을 이용해서 프로그램을 다운로드 할 수 있습니다.

- ◆ 프로젝트 메뉴 >> 옵션을 선택하고 접속 옵션 탭을 선택하여 다음과 같이 설정합니다.




- ◆ 리모트 접속의 경우 통신 종류에 따라 사용 방법 및 파라미터 설정 방법이 다르므로 자세한 내용은 각 통신 모듈 사용 설명서를 참조하십시오.

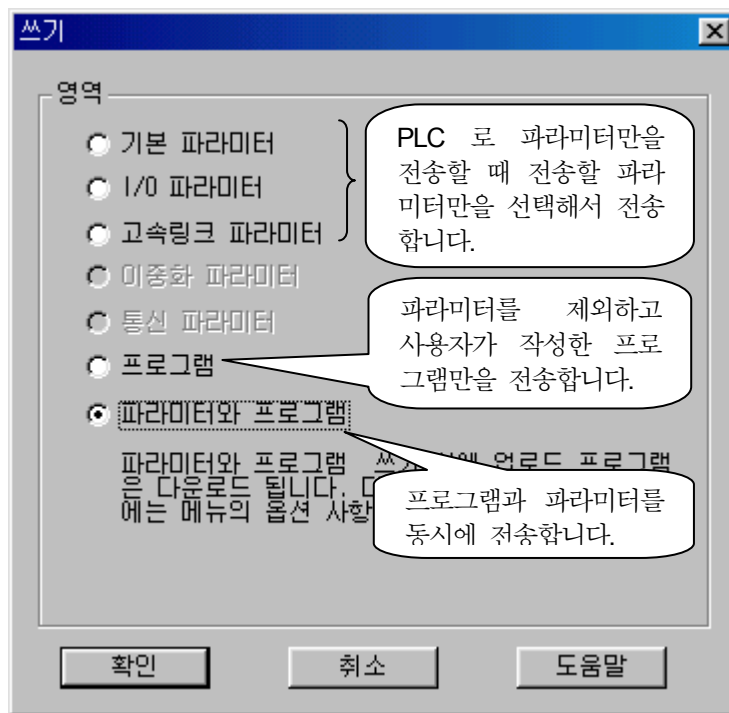
- ◆ 접속이 완료되면 **결과창**의 **통신** 항목에 접속되었다는 메시지가 다음과 같이 타납니다.



4.1.4.3 쓰기

PC 에서 작성한 프로젝트를 PLC 의 프로그램 메모리 영역에 써 넣는 작업을 **쓰기**라고 합니다.

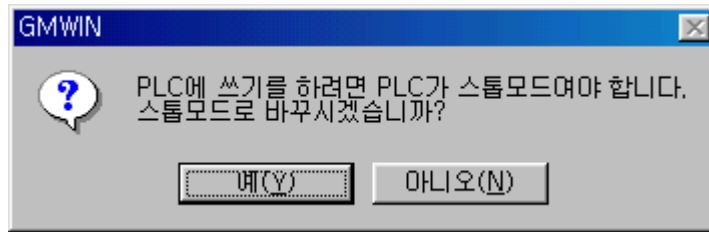
- ◆ 온라인 메뉴 >> 쓰기를 클릭하거나 쓰기 아이콘()을 클릭하면 다음과 같은 화면이 나타납니다.



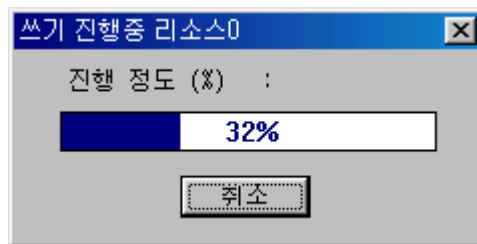
- ◆ **파라미터와 프로그램**을 선택한 후 **확인**을 클릭합니다.

알아두기 쓰기를 하면 실행 프로그램과 업-로드 프로그램이 동시에 전송됩니다. 만일, 업-로드 프로그램을 전송하지 않으려면 프로젝트 메뉴의 메이크 옵션에서 업-로드 프로그램을 선택하지 않고 컴파일 및 메이크를 실행한 후 쓰기를 하면 업-로드 프로그램은 CPU 로 전송되지 않으며, CPU 에 들어있는 프로그램을 PC 로 불러올 수 없습니다.

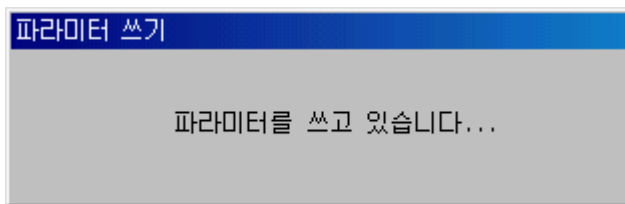
- ◆ CPU 의 모드가 리모트 모드이면서 RUN 상태이면 다음과 같은 메시지가 나타납니다. 여기서 예를 클릭하면 CPU 가 STOP 상태로 전환되면서 다운로드가 실행됩니다.



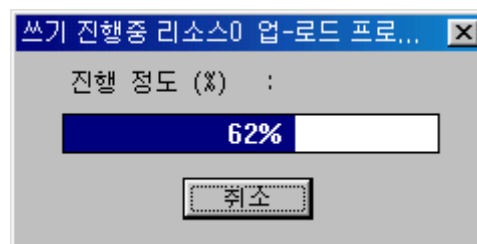
- ◆ 다운로드가 실행되면 제일 먼저 실행 파일이 CPU 로 전송됩니다.



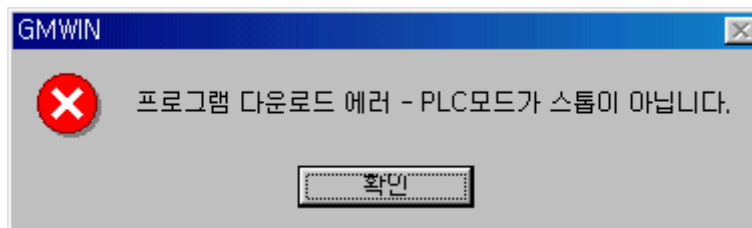
- ◆ 실행 파일 전송 완료 후 파라미터가 전송됩니다.



- ◆ 파라미터 전송 완료 후 업-로드 프로그램이 전송됩니다.




- ◆ CPU 가 로컬 런 상태이면 다음과 같은 메시지가 나옵니다. 이 경우 **확인**을 클릭한 후 CPU 를 리모트 모드로 전환하면 프로젝트를 전송할 수 있습니다.



4.1.5 모드 전환 및 모니터


4.1.5.1 모드 전환

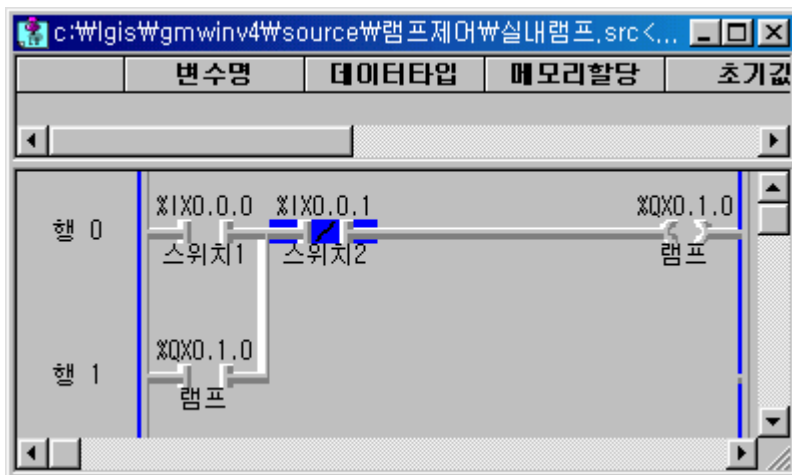
전송이 완료되었으면 온라인 메뉴 >> 모드 전환 >> 런(R)을 클릭하거나 RUN 단축 아이콘()을 클릭하여 CPU 를 RUN 모드로 전환합니다.
CPU 가 RUN 모드로 전환되면 CPU 는 프로그램 연산을 실행합니다.

4.1.5.2 모니터 시작

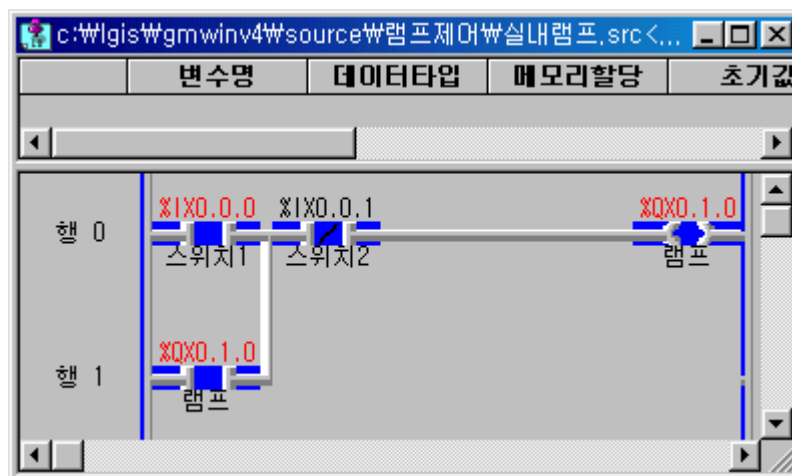
1) 래더 프로그램에서 모니터

PLC 가 운전 중에 프로그램이 작성할 때의 의도대로 동작하는지 여부와 데이터가 변하는 것을 GMWIN 을 이용하여 확인해 볼 수 있습니다.

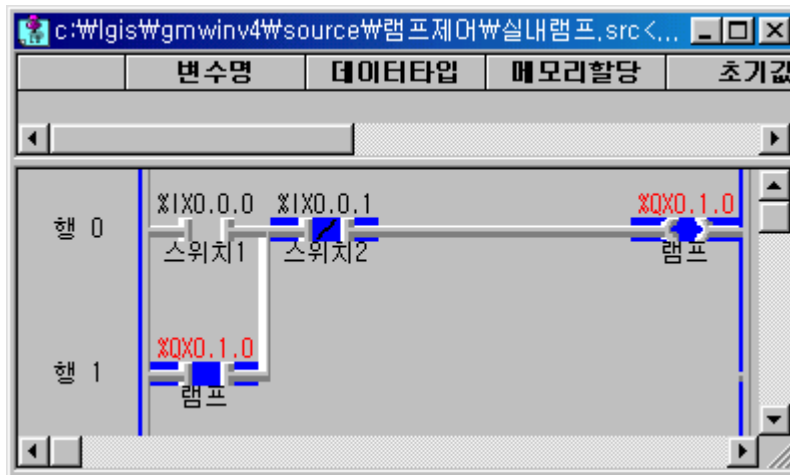
- ◆ 온라인 메뉴 >> 모니터 시/끝(M)을 클릭하거나 모니터 시작 단축 아이콘()을 클릭하면 래더 프로그램에서 입력 및 출력 상태를 볼 수 있습니다.



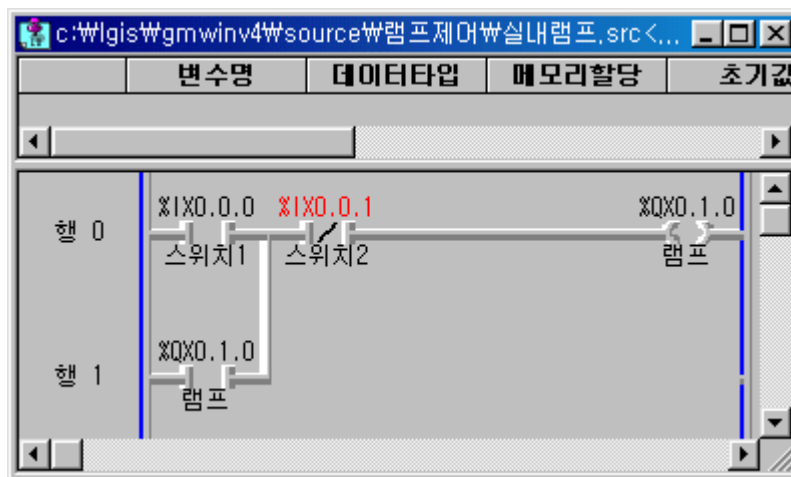
< 초기의 모니터 화면 >



< %IX0.0.0 을 ON 시켰을 때 >




< %IX0.0.0 을 OFF 시켰을 때 >



< %IX0.0.1 을 ON 시켰을 때 >

2) 변수 모니터

프로그램에 등록된 변수를 일괄 선택하여 데이터가 변하는 모습을 모니터링 하거나 특정 변수들을 선택하여 선택된 변수 만을 모니터링 또는 값을 변경 할 수 있습니다.

◆ 온라인 메뉴 >> 모니터 시작/끝에서 모니터를 기동시킨 상태에서 변수 모니터 창 단축 아이콘 ()을 클릭하면 프로젝트에 등록된 변수를 모니터 할 수 있습니다.

변수

컨피그레이션

글로벌 변수

리소스


글로벌 변수

인스턴스 변수

직접 변수


플래그

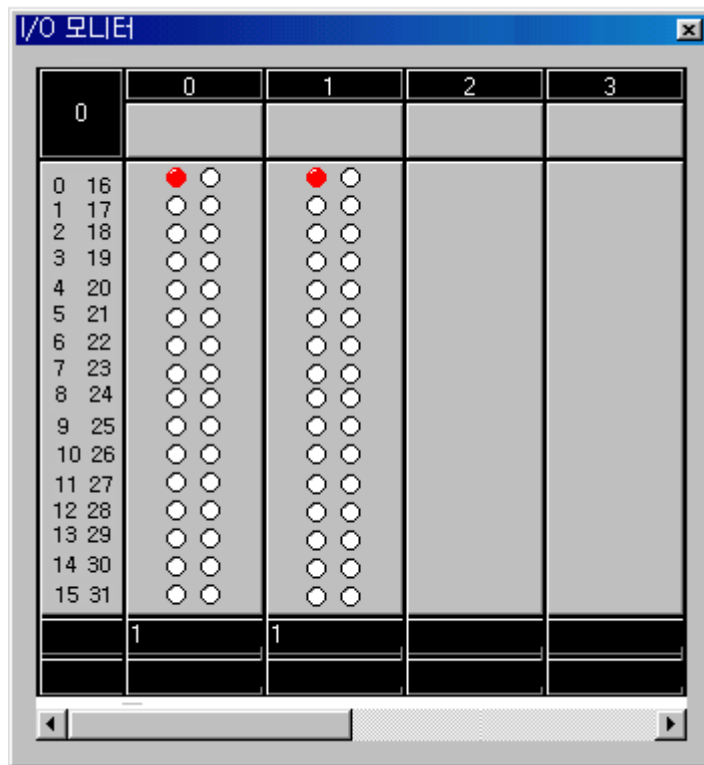
리소스	인스턴스	변수명	변수값
직접 변수		%IX0.0.0	1
직접 변수		%IX0.0.1	0
직접 변수		%QX0.1.0	1

알아두기 온라인 메뉴 >> 접속 + 쓰기 + 모드 전환(런) + 모니터 시작을 클릭하거나 단축 아이콘 ()을 클릭하면 접속, 쓰기, 모드 전환 및 모니터 시작이 일괄적으로 실행됩니다.

3) I/O 모니터

GMWIN 을 이용해서 PLC 의 입력 및 출력 상황을 볼 수 있습니다.

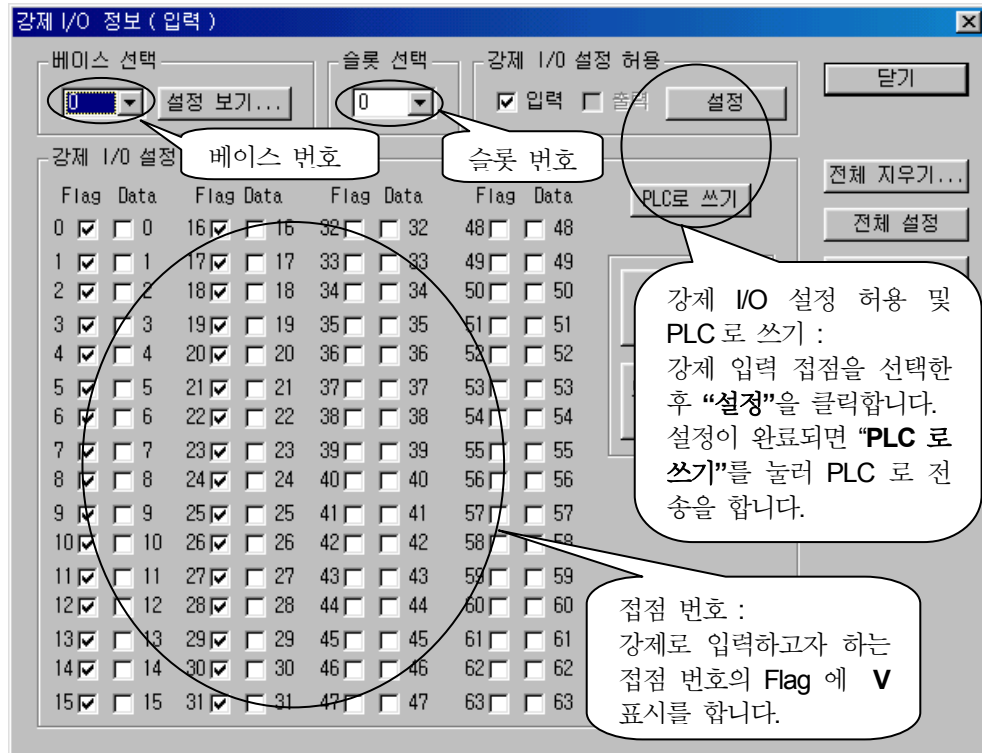
- ◆ 온라인 메뉴 >> 모니터 시작/끝에서 모니터를 기동시킨 상태에서 I/O 모니터 창 단축 아이콘()을 클릭하면 PLC 의 입력 및 출력 상황을 볼 수 있습니다.



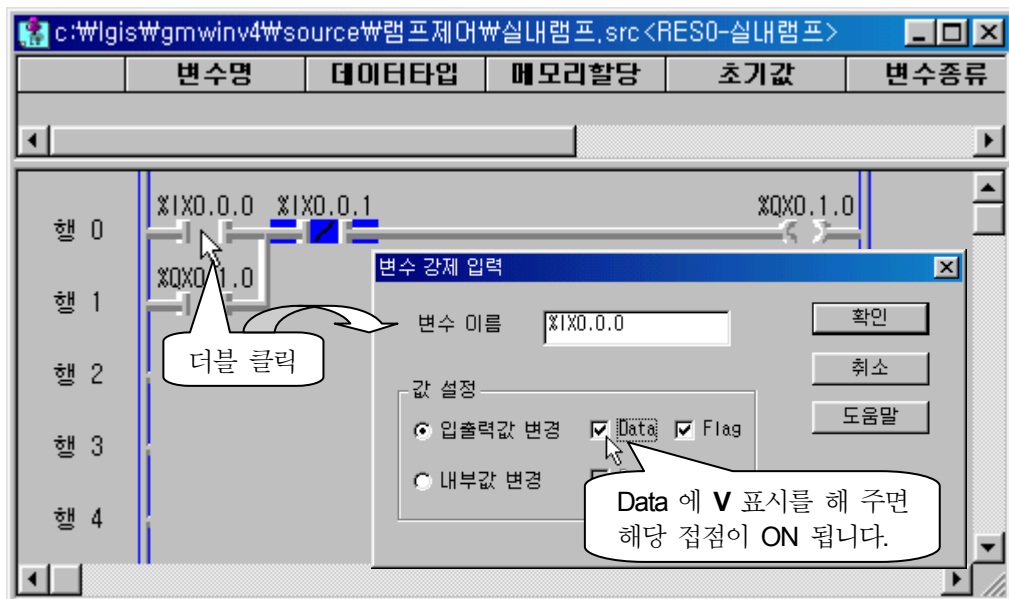
4.1.6 강제 I/O 설정

PLC 프로그램 작성 완료 후 PLC 본체는 있지만 입력 및 출력 배선이 되어있지 않은 경우, GMWIN 에서 강제 I/O 설정 기능을 이용하여 입력 신호를 주어 프로그램을 디버깅 해 볼 수 있습니다.

- ◆ PLC 와 GMWIN 이 접속이 된 상태에서 온라인 메뉴 >> 강제 I/O 설정 >> 입력을 클릭 하면 다음과 같은 강제 I/O 설정 창이 나타납니다.



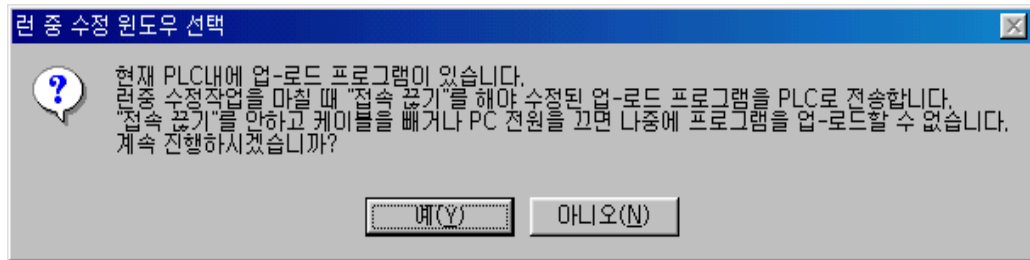
- ◆ 모니터링 상태에서 점점을 더블 클릭하면 데이터를 강제로 ON 또는 OFF 할 수 있는 창이 나타납니다.



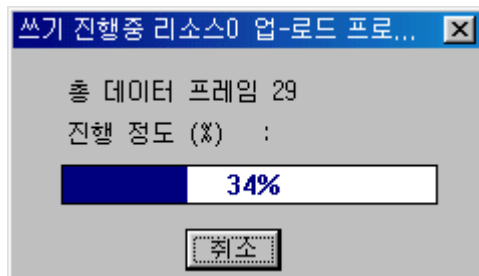
4.1.7 런 중 수정

GLOFA-GM 시리즈 PLC 는 CPU 가 RUN 상태에서 프로그램을 수정할 수 있습니다.

- ◆ PLC 에 다운로드 된 프로그램을 GMWIN 화면에 열어 놓은 상태에서 **온라인 메뉴 >> 런 중 편집 >> 런 중 수정 시작**을 클릭하면 런 중 수정 모드로 전환됩니다.
이 때, PLC 에 업-로드 프로그램이 전송되어 있는 경우 다음과 같은 메시지가 나타나는데 “ 예(Y)” 를 클릭하면 런 중 수정 모드로 전환됩니다.
이 때 PLC 의 CPU 는 RUN 상태를 유지합니다.



- ◆ 프로그램의 수정이 끝나면 온라인 메뉴의 ‘ 런 중 쓰기’ 를 클릭하면 수정된 프로그램을 PLC 로 전송합니다. 이 경우에도 PLC 의 CPU 는 RUN 상태를 유지합니다.
- ◆ 런 중 수정된 프로그램을 업-로드하기 위해서는 다운로드 케이블을 PLC 로부터 분리하기 전에 반드시 온라인 메뉴의 ‘ 접속 끊기’ 를 선택하여 PC 가 가지고 있는 수정된 업-로드 정보를 PLC 로 전송해야 합니다.
만일, 접속 끊기를 실행하지 않고 다운로드 케이블을 PLC 로부터 분리했을 경우, 업-로드를 할 수 없습니다.



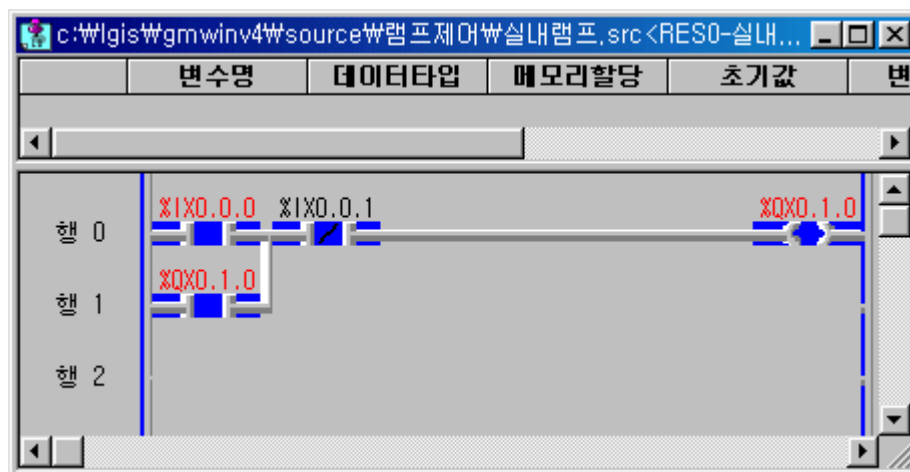
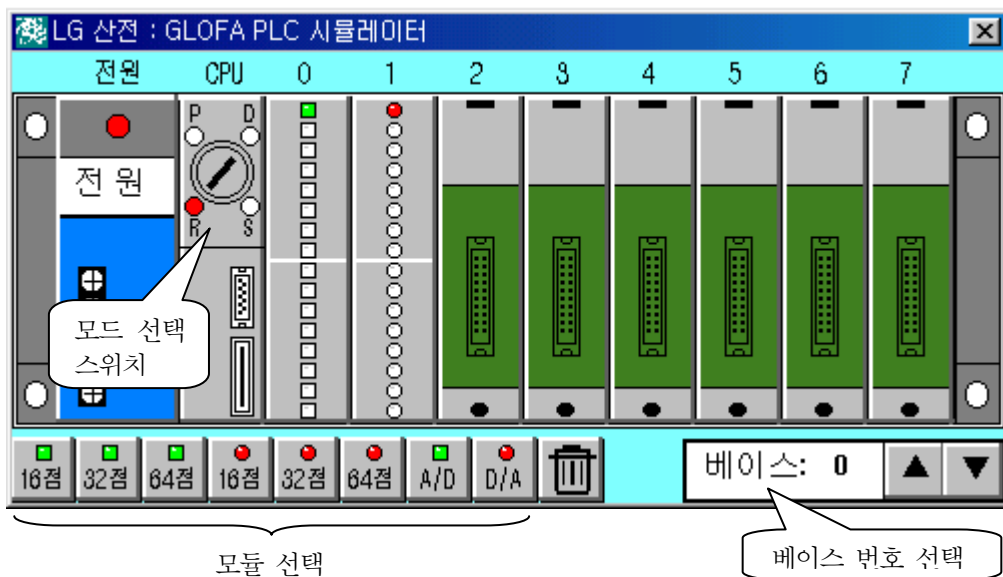
4.1.8 시뮬레이터

GMWIN 은 PLC 가 없어도 가상 운전을 해 볼 수 있도록 시뮬레이터 기능을 내장하고 있어 미리 프로그램의 동작 상황을 검사해 볼 수 있습니다.

◆ 프로그램 작성이 완료되면 **도구 메뉴 >> 시뮬레이터 시작**을 클릭하면 컴파일을 실행하고 PLC 모양이 화면에 나타납니다.

시뮬레이터가 실행될 때는 CPU 의 모드가 STOP 모드(S)로 되어있습니다.

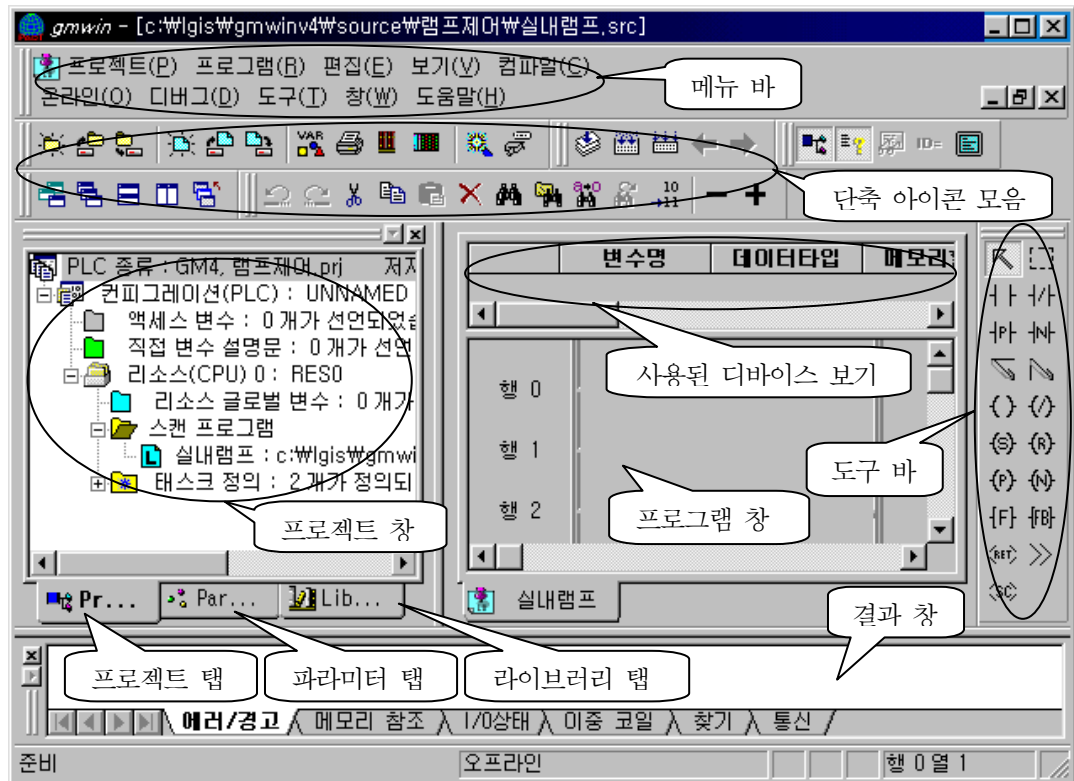
모드를 RUN(R)으로 선택한 후 시뮬레이터에 있는 입력 접점을 클릭하면 해당하는 접점이 ON 되며 프로그램 상에도 모니터링 표시가 됩니다.



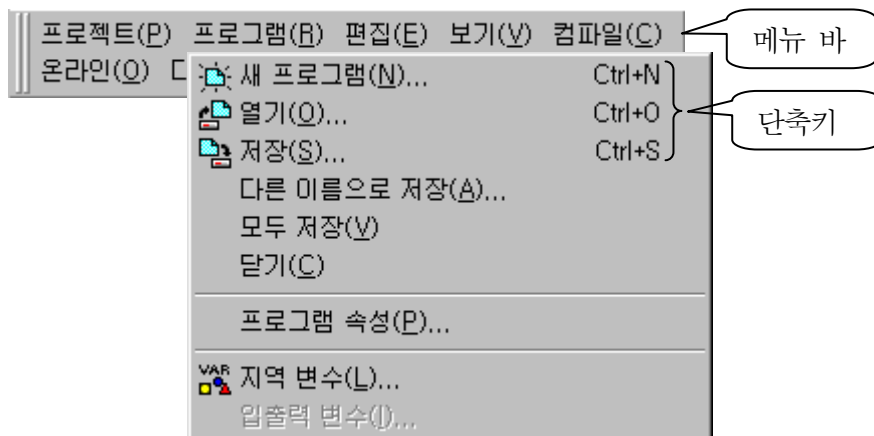
알아두기 시뮬레이션은 PLC 가 OFF-LINE 상태에서에서만 실행이 가능합니다.
PLC 가 접속된 상태이면 접속을 해제하고 시뮬레이터를 기동해야 합니다.
시뮬레이션은 DI, DO, AD, DA 모듈만 가능합니다.

4.2 화면 구성

GMWIN 화면은 아래와 같은 구성으로 이루어져 있습니다.



4.2.1 메뉴



메뉴를 선택하면 명령들이 나타나고, 원하는 명령을 마우스 또는 키로 선택하면 명령을 실행할 수 있습니다.

생략기호(...)가 붙은 명령을 선택하면 하위의 대화상자가 나타납니다.

단축키(Ctrl+X, Ctrl+C...)가 있는 메뉴인 경우에는 단축키를 눌러서 직접 명령을 선택할 수 있습니다.

4.2.1.1 프로젝트

명 령	설 명
새 프로젝트	프로젝트를 처음 생성합니다.
열기	기존의 프로젝트를 엽니다.
PLC 로 부터 열기	PLC 에 있는 프로젝트 및 프로그램을 업-로드 합니다.
저장	프로젝트를 저장합니다. 프로그램은 저장되지 않습니다.
다른 이름으로 저장	프로젝트를 다른 이름으로 저장합니다.
닫기	프로젝트를 닫습니다.
프로젝트 묶음 열기	프로젝트 묶음 파일을 엽니다.
프로젝트 묶음 만들기	프로젝트에 연결된 모든 파일들을 하나의 파일로 묶어 줍니다.
프로젝트 항목 추가	프로젝트에 새로운 항목(프로그램 정의, 리소스, 태스크, 라이브러리, 리소스는 GM1 만 해당)을 추가합니다.
M 영역 수정	M 영역을 편집하거나 저장하도록 합니다.
미리보기	인쇄될 화면을 미리 보여 줍니다.
인쇄	활성화되어 있는 창의 내용을 인쇄합니다.
프린터 설정	프린터 옵션을 설정합니다.
옵션	GMWIN 에 해당되는 옵션을 설정합니다.
이전 프로젝트 목록	이전에 작업한 프로젝트를 열어 줍니다.
종료	GMWIN 을 끝마칩니다.

4.2.1.2 프로그램

명 령	설 명
새 프로그램 Ctrl+N	프로그램을 처음 생성 합니다.
열기 Ctrl+O	기존의 프로그램을 엽니다.
저장 Ctrl+S	프로그램을 저장 합니다.
다른 이름으로 저장	프로그램을 다른 이름으로 저장 합니다.
닫기	프로그램을 닫습니다.
프로그램 속성	프로그램의 속성을 바꿉니다.
지역 변수	변수를 편집 합니다.
입출력 변수	평선, 평선 블록인 경우 입출력 변수를 편집 합니다.
이전 프로그램 목록	이전에 작업한 프로그램을 엽니다.

SFC 인 경우 추가

명 령	설 명
액션 목록	SFC 인 경우 액션 목록을 봅니다.
트랜지션 목록	SFC 인 경우 트랜지션 목록을 봅니다.
SFC 속성	SFC 속성들을 지정합니다.

4.2.1.3. 편집

명 령	설 명
편집 취소 Ctrl+Z	프로그램 편집 창에서 편집을 취소하고 바로 이전상태로 되돌립니다.
재실행 Ctrl+Y	편집 취소된 동작을 다시 복구합니다.
잘라 내기 Ctrl+X	블록을 잡아 삭제하면서 클립보드에 복사합니다.
복사 Ctrl+C	블록을 잡아 클립보드에 복사합니다.
붙여 넣기 Ctrl+V	클립보드로부터 편집 창에 복사합니다.
삭제 Del	블록을 잡아 삭제합니다.
찾기 Ctrl+F	원하는 문자를 찾습니다.
바꾸기 Ctrl+H	원하는 문자를 찾아 새로운 문자로 바꿉니다.
다시 찾기 Ctrl+F3	이전에 실행한 찾기(Find) 또는 바꾸기(Replace)를 반복 실행 합니다.
찾아 가기	원하는 위치로 커서를 이동합니다.
여러 파일에서 찾기	프로젝트 또는 원하는 경로에 있는 모드 파일에서 문자를 찾습니다.
편집 도구	각 프로그램에 사용되는 편집 도구들이 있습니다.

LD 인 경우 추가

명 령	설 명
라인 삭제 Ctrl+D	한 줄을 지웁니다.
라인 삽입 Ctrl+L	한 줄을 삽입합니다.
셀 삽입 Ctrl+I	한 셀을 삽입합니다.

각 프로그램별 편집 도구

IL 편집 시

명 령	설 명
평선 F2	평선 삽입
평선 블록 F3	평선 블록 삽입
레이블 F4	레이블 삽입
오퍼레이터 F5	연산자 삽입

LD 편집 시

메뉴에 해당되는 점점, 코일, 평선, 평선 블록, 점프, 리턴 등을 삽입 합니다.

SFC 편집 시

명 령	설 명
스텝 F2	스텝/트랜지션 삽입
분기 F3	병렬 또는 선택 분기 삽입
액션/트랜지션 F4	액션 또는 트랜지션 이름 삽입
레이블 F5	레이블 삽입
점프 F6	점프 삽입
중 F7	액션 또는 트랜지션에 들어가서 프로그램을 편집합니다.

4.2.1.4 보기

명 명	설 명
도구모음	도구상자를 사용자가 정의하도록 합니다.
상태 표시줄	상태 표시줄을 보이거나 숨깁니다.
전체화면	프로그램 창이 표시될 영역을 화면 전체로 확대 합니다.
프로젝트 창	프로젝트 창을 보이거나 숨깁니다.
결과 창	결과 창을 보이거나 숨깁니다.
변수 모니터 창	변수 모니터 창을 보이거나 숨깁니다.
I/O 모니터 창	I/O 모니터 창을 보이거나 숨깁니다.
링크 파라미터 창	링크 파라미터 창을 보이거나 숨깁니다.
확대/축소	화면을 확대 또는 축소합니다.
변수 설명문	변수 설명문을 보이거나 숨깁니다.
등록 정보	현재 선택된 항목의 등록 정보를 보입니다.

LD 편집 시

명 명	설 명
확대/축소	LD 화면을 확대 또는 축소 합니다.
메모리 위치/설명문	변수의 메모리와 설명문을 보이거나 숨깁니다.

SFC 편집 시

명 명	설 명
화면 확대/축소	SFC 화면을 확대 또는 축소 합니다.
설명문 보이기	변수 설명문을 보이거나 숨깁니다.
액션 감추기	액션을 보이거나 숨깁니다.

4.2.1.5 컴파일

명 명	설 명
컴파일	프로그램을 컴파일 합니다.
메이크	프로젝트에 속해 있는 프로그램 중 컴파일이 안된 프로그램들을 컴파일 한 후 PLC 실행 파일을 만듭니다.
모두 컴파일	프로젝트에 속해 있는 모든 프로그램을 컴파일 한 후 PLC 실행 파일을 만듭니다.
메모리 참조	사용된 글로벌 변수 및 직접 변수를 볼 수 있습니다.
I/O 사용 상태	입/출력 직접 변수의 사용 현황을 도표를 보여 줍니다.
이중 코일 검사	사용된 이중 코일을 보여 줍니다.
이전 메시지로	이전 메시지 위치로 이동 합니다.
다음 메시지로	다음 메시지 위치로 이동 합니다.

4.2.1.6 온라인

명 령		설 명
접속+쓰기+모드전환(런)+ 모니터시작(G) Ctrl+R		GMWIN 과 옵션에서 지정한 PLC 를 접속시켜 사용자가 작성 한 프로그램을 PLC 에 쓴 후 모드를 전환하여 모니터링 합니다.
접속		GMWIN 과 옵션에서 지정한 PLC 를 접속시킵니다.
접속 끊기		GMWIN 과 PLC 접속을 해제합니다.
읽기		PLC 의 데이터를 읽어 옵니다.
쓰기		GMWIN 의 프로그램을 PLC 에 씁니다.
모니터 시작/끝		프로그램을 모니터링 합니다./모니터링을 끝냅니다.
모드 변환	런	PLC 모드를 전환합니다.
	스톱	
	일시 정지	
	디버그	
	마스터 전환	GM1 에서 통신할 CPU 를 전환합니다.
리셋	데이터 클리어	PLC 데이터를 0 으로 지웁니다.
	리셋	PLC 를 리셋 합니다.
	Overall 리셋	
플래시 메모리	타입 정보	CPU 에 장착된 플래시 메모리의 타입 정보를 읽거나 플래시 메모 리에 데이터 쓰기를 합니다.
	쓰기	
PLC 정보	시스템 정보	PLC 정보를 보여줍니다.
	에러/경고 상세 정보	
	PLC 이력	
	입/출력 고장 상세 정보	
I/O 설정	I/O 정보	PLC I/O 구성 상태를 보이고 씁니다
	I/O 동기화	PLC I/O 구성을 프로젝트와 PLC 에서 일치 시킵니다.
강제 I/O 설정	입력	I/O 강제 입출력 값/실행 허용을 설정 합니다.
	출력	
네트워크	링크 허용설정	링크 모듈의 타입,장착 슬롯,국번 등을 보여 줍니다.
	네트워크 정보	네트워크 정보를 봅니다.
	Mnet 파라미터	Mnet 파라미터를 입력합니다.
런 중 편집	수정 시작	런 중 편집을 시작합니다.
	쓰기	런 중 편집된 내용을 씁니다
	취소	런 중 편집을 취소합니다.
FSM		F-net 슬레이브 모듈의 비상 데이터를 설정합니다.
I/O 스킵		스킵 할 I/O 를 지정합니다.
고장 마스크 설정		고장 마스크를 설정합니다.
특수 모듈 초기화		특수 모듈을 초기화 합니다.

4.2.1.7 디버그

명 령	설 명
디버그 시작/끝	디버그 모드로 전환하여 디버그를 시작합니다/디버그를 끝냅니다.
런 Ctrl+F9	브레이크 포인트까지 런 시킵니다.
스텝 오버 Ctrl+F8	한 스텝씩 런 시킵니다.
스텝 인	평선, 평선 블록을 디버깅합니다.
스텝 아웃	평선, 평선 블록 디버그 시 현재 블록을 빠져 나갑니다.
일시 정지	런을 중지시킵니다.
커서 위치까지 런 Ctrl+F2	커서 위치까지 런 시킵니다.
브레이크 포인트 설정/해제 Ctrl+F5	브레이크 포인트를 설정 또는 해제합니다.
브레이크 포인트 목록 / 조건	설정된 브레이크 포인트의 목록을 보여주고 브레이크 조건을 설정합니다.
태스크 수행 설정	디버깅 중 태스크 전환을 허용합니다.

4.2.1.8 도구

명 령	설 명
라이브러리 관리자	라이브러리를 편집합니다.
시뮬레이터 시작	시뮬레이터를 시작합니다.
데이터 공유	모니터 값들을 엑셀과 공유합니다.

4.2.1.9 창

명 령	설 명
새 창	현재 창에 대해 새 창을 엽니다.
계단식 배열	GMWIN에 속해 있는 여러 창들을 계단식으로 배열합니다.
수평 배열	GMWIN에 속해 있는 여러 창들을 수평 배열합니다.
수직 배열	GMWIN에 속해 있는 여러 창들을 수직 배열합니다.
아이콘 정렬	GMWIN에 속해 있는 아이콘들을 정렬합니다.
모두 닫기	GMWIN에 속해 있는 여러 창들을 모두 닫습니다.

4.2.1.10 도움말

명 령	설 명
GMWIN 사용 도움말	GMWIN 도움말을 엽니다.
도움말 사용법	도움말 사용법을 엽니다.
LG 산전 홈 페이지	LG 산전 홈 페이지에 인터넷 접속합니다.
LG 산전 GMWIN 정보	GMWIN의 정보를 나타냅니다.

4.2.2 도구모음

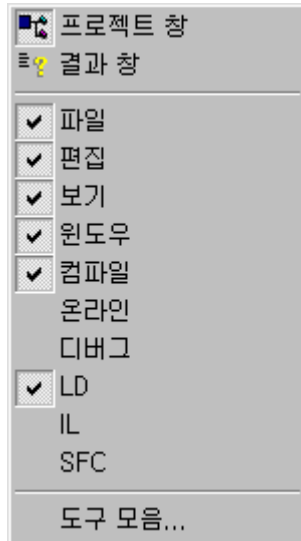
GMWIN에서는 현재 자주 사용되는 메뉴들을 단축 형태인 도구로 제공하고 있습니다. 원하는 도구를 마우스로 누르면 실행 됩니다. 아래 표에서는 도구의 모양과 그에 대한 설명을 나타냅니다.



도구	명 령	도구	명 령	도구	명 령
	새 프로젝트		접속+쓰기+모드전환(런)+모니터 시작		실행파일 만들기
	프로젝트 열기		접속		라이브러리 관리자
	프로젝트 저장		접속 끊기		시뮬레이터
	새 프로그램		쓰기		재실행
	프로그램 열기		모니터 시작/끝		여러 파일에서 찾기
	프로그램 저장		런		찾아 가기
	지역 변수		스톱		수직 배열
	편집 취소		일시 정지		모두 닫기
	잘라내기		디버그 시작		프로젝트 창
	복사		디버그 런		결과 창
	붙여넣기		스텝오버		변수 모니터 창
	삭제		스텝인		I/O 모니터 창
	찾기		스텝아웃		화면 축소
	바꾸기		일시정지		화면 확대
	다시 찾기		커서 위치까지 런		인쇄
	컴파일		브레이크 포인트 설정/해제		새 창
	전체 화면		런 중 쓰기		계단식 배열
	이전 메시지		시스템 정보		수평 배열
	다음 메시지		I/O 정보		PLC 이력
	런 중 수정 시작		데이터 공유		

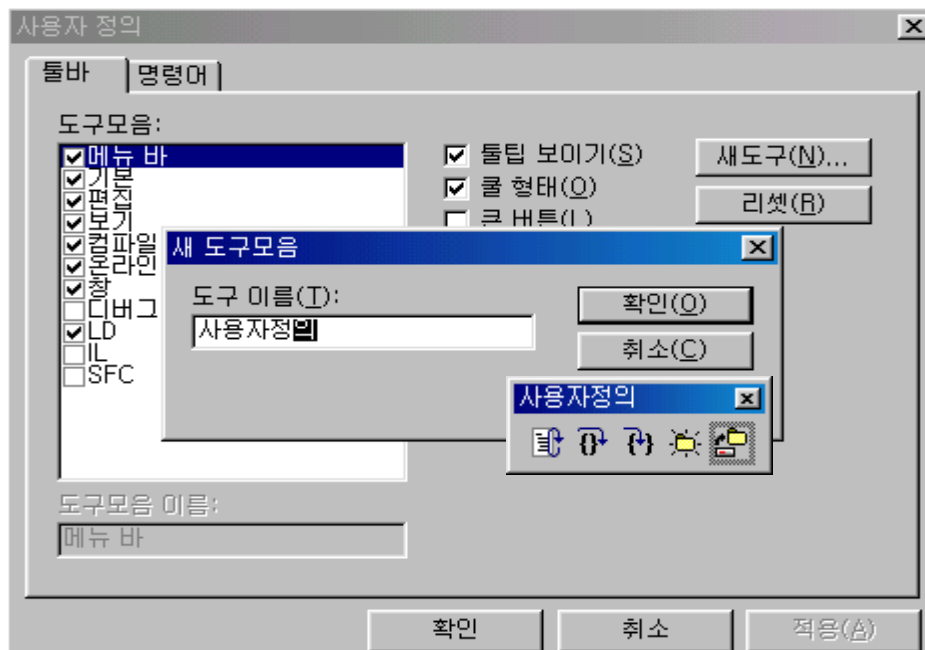
알아두기 도구 모음 추가 및 삭제하기

- ◆ 윈도우의 도구 모음에서 마우스의 오른쪽 버튼을 눌러 팝업 메뉴를 부릅니다.
- ◆ 팝업 메뉴에서 추가 또는 삭제를 원하는 메뉴를 선택합니다.

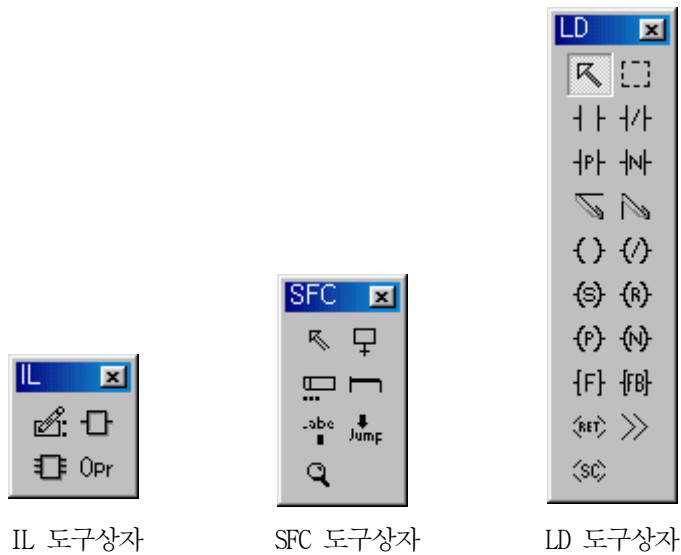


도구 모음 사용자 정의

- ◆ 메뉴 [보기]-[도구모음]-[새도구] 선택합니다.
- ◆ 새도구 이름 입력후 원하는 아이콘을 새로 만들어진 도구 모음에 올려 놓고 원하는 위치에 도킹 시킵니다.



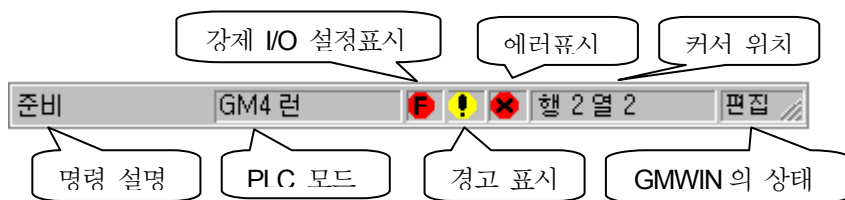
4.2.3 도구상자



프로그램 편집 할 때 자주 사용하는 명령을 도구상자를 통해서 실행할 수 있습니다. 도구를 마우스로 누르면 도구가 실행합니다. 설정되어 있는 도구들은 메뉴 **편집 >> 편집도구**를 통해서도 실행될 수 있습니다.

메뉴 **보기 >> 도구모음**을 선택하거나 팝업 메뉴를 이용해 도구상자의 위치와 화면 상에 나타나는 모양을 조정할 수 있습니다.

4.2.4 상태 표시 줄



4.2.4.1 명령 설명

반전 표시된 메뉴나 명령, 마우스가 위치해 있는 도구모음에 대한 설명을 나타냅니다.


4.2.4.2 PLC 모드 표시

PLC의 모드를 나타냅니다.


PLC와 연결되지 않았을 때에는 오프라인으로 표시됩니다.

오프라인-런-스톱-일시 정지-디버그


4.2.4.3 강제 입출력 표시

강제 입력 또는 출력을 설정한 경우  로 표시됩니다.

4.2.4.4 경고 표시

PLC 에 이상상태(경고)가 발생한 경우  로 표시됩니다.

4.2.4.5 에러 표시

PLC 에 이상상태(에러)가 발생한 경우  로 표시됩니다.

4.2.4.6 커서 위치 표시

프로그램을 편집할 때 커서의 위치를 나타냅니다.

4.2.4.7 GMWIN 상태 표시

GMWIN 의 상태를 표시합니다.

편집 : GMWIN 에서 프로그램을 편집중임을 나타냅니다.

모니터 : PLC 의 데이터를 모니터링 중임을 나타냅니다.

디버그 : PLC 의 프로그램을 디버깅 중임을 나타냅니다.

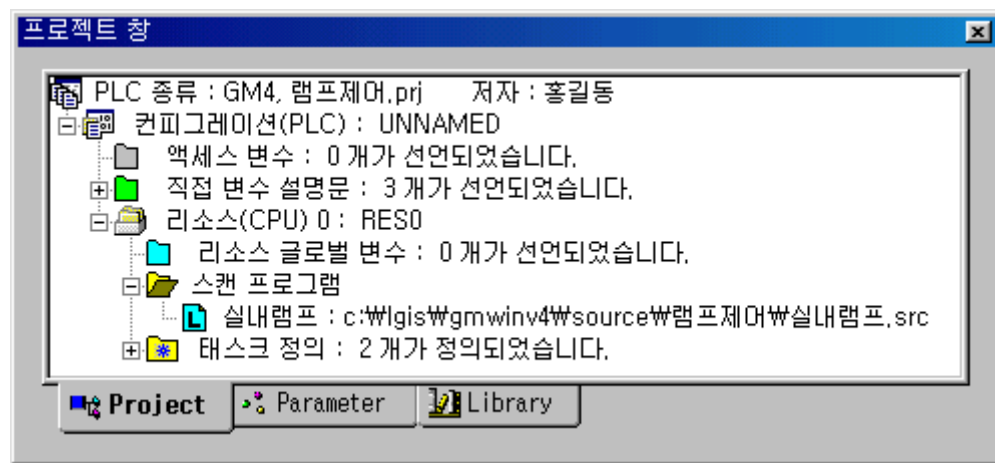
4.3 프로젝트의 구조

프로젝트는 GLOFA-GM PLC 의 시스템을 구성하는 가장 기본적인 요소로서 한 PLC 시스템 당 하나의 프로젝트를 작성함을 기본으로 합니다. 프로젝트는 크게 프로젝트 탭(컨피그레이션), 파라미터 탭, 라이브러리 탭으로 나누어져 있습니다.

프로젝트 탭에서는 액세스 변수, 직접 변수 설명문, 리소스 내용 등 소프트웨어적인 것들을 작성하고, 파라미터 탭에서는 기본 파라미터, I/O 파라미터, 링크 파라미터 등 하드웨어적인 것들을 작성하는 부분입니다.

그리고, 라이브러리 탭에서는 라이브러리 파일을 추가, 삭제할 수 있습니다.

.프로젝트는 다음과 같은 계층 구조를 가지고 있습니다.



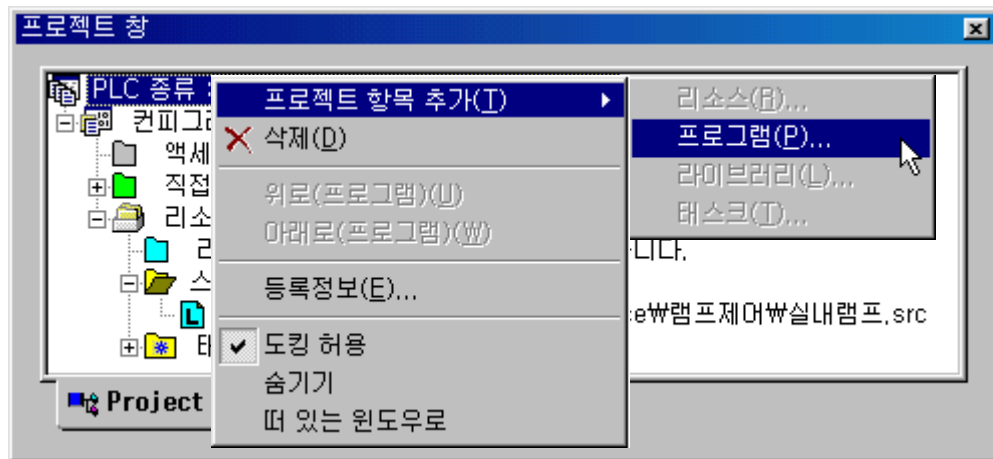
계 층 항 목	설 명
프로젝트 탭	PLC 시스템 전체를 정의
컨피그레이션	PLC 프로그램에 관한 여러 정의 사항들을 설정
액세스 변수	다른 컨피그레이션이 접근 가능한 변수 리스트
직접 변수 설명문	직접 변수에 사용한 설명문 리스트
리소스	CPU 모듈에 해당
리소스 글로벌 변수	한 리소스 전체에서 사용되는 변수 리스트
스캔 프로그램	스캔 프로그램을 정의
태스크 정의	프로그램의 실행 조건 정의
파라미터 탭	PLC 시스템의 하드웨어에 관한 내용 정의
기본 파라미터	기본적인 하드웨어 파라미터 정의
I/O 파라미터	입출력 모듈에 관한 내용 기술
고속 링크 파라미터	고속 링크 파라미터에 관한 내용 기술
라이브러리 탭	현재 삽입되어 있는 라이브러리 파일들의 리스트
평선 라이브러리	프로젝트에 등록된 평선 라이브러리 리스트
평선 블록 라이브러리	프로젝트에 등록된 평선 블록 라이브러리 리스트

4.3.1 프로젝트

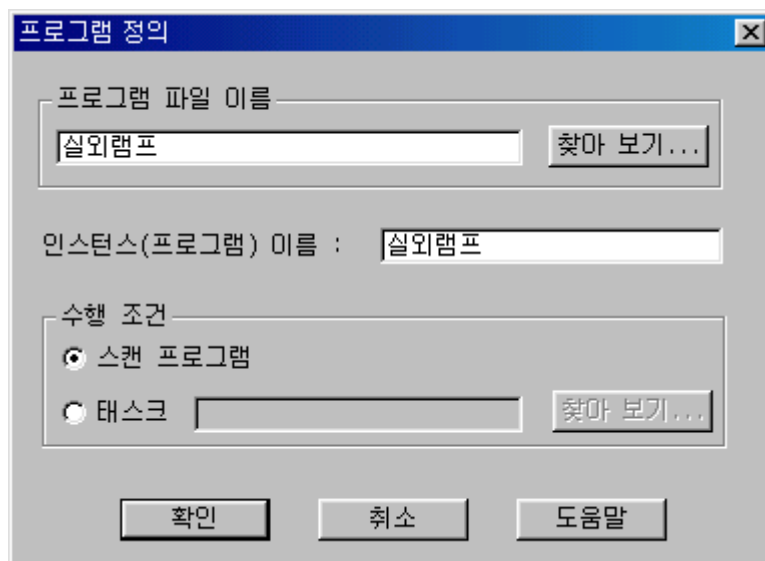
4.3.1.1 프로그램 추가

GLOFA-GM 시리즈 PLC 는 하나의 프로젝트에 여러 개의 프로그램을 넣을 수 있습니다. 하나의 PLC 를 이용하여 여러 가지 기기를 제어할 때 제어 대상에 따라 프로그램을 나누어 작성하면 프로그램을 간단하게 작성할 수 있으며, 디버깅할 때 제어 기기들의 오동작 여부를 보고 오동작하는 제어기기에 관한 프로그램만 수정할 수 있으므로 디버깅의 절차가 간단해 질 수 있습니다.

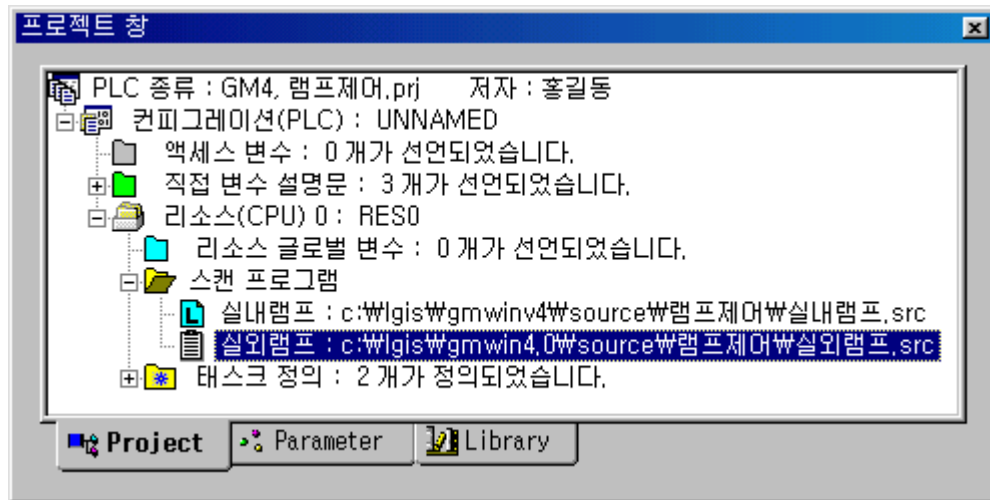
- ◆ 프로젝트 창에서 마우스의 오른쪽 키를 눌러 **프로젝트 항목 추가 >> 프로그램**을 선택하여 새로운 “프로그램 정의” 창을 불러냅니다.



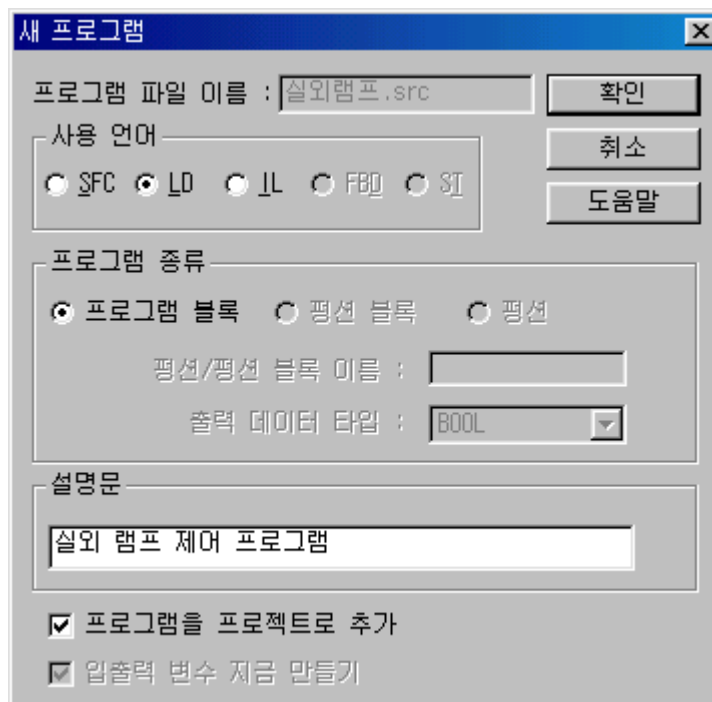
- ◆ 프로그램 파일 이름과 새로운 인스턴스 이름을 등록합니다.



- ◆ 프로젝트 탭의 스캔 프로그램에 새로운 프로그램의 인스턴스가 등록되었습니다.



- ◆ 새로 등록된 프로그램 인스턴스를 더블 클릭하여 프로그램에서 사용할 언어와 설명문을 입력한 후 '확인'을 누르면 새로운 프로그램 창이 나타납니다.



4.3.1.2 글로벌 변수

앞에서 설명한 바와 같이 GLOFA-GM 시리즈 PLC 는 하나의 프로젝트에 여러 개의 프로그램을 입력할 수 있으며, 하나의 프로그램에서 사용한 변수를 글로벌 변수로 등록하면 동일 프로젝트 내의 여러 개의 프로그램에서 공통으로 사용할 수 있습니다.

- ◆ 글로벌 변수로 사용하기 위해서는 최초 변수를 등록할 때 변수의 종류를 VAR_EXTERNAL 로 설정합니다.

변수명	데이터타입	메모리할당	초기값	변수종류	사용여!
1 카운터	INT	<자동>		VAR	*
2 OFF	BOOL	<자동>		VAR_EXTERNAL	*

변수 추가/수정

변수 이름 : OFF

변수 종류 : VAR_EXTERNAL

데이터 타입 : BOOL

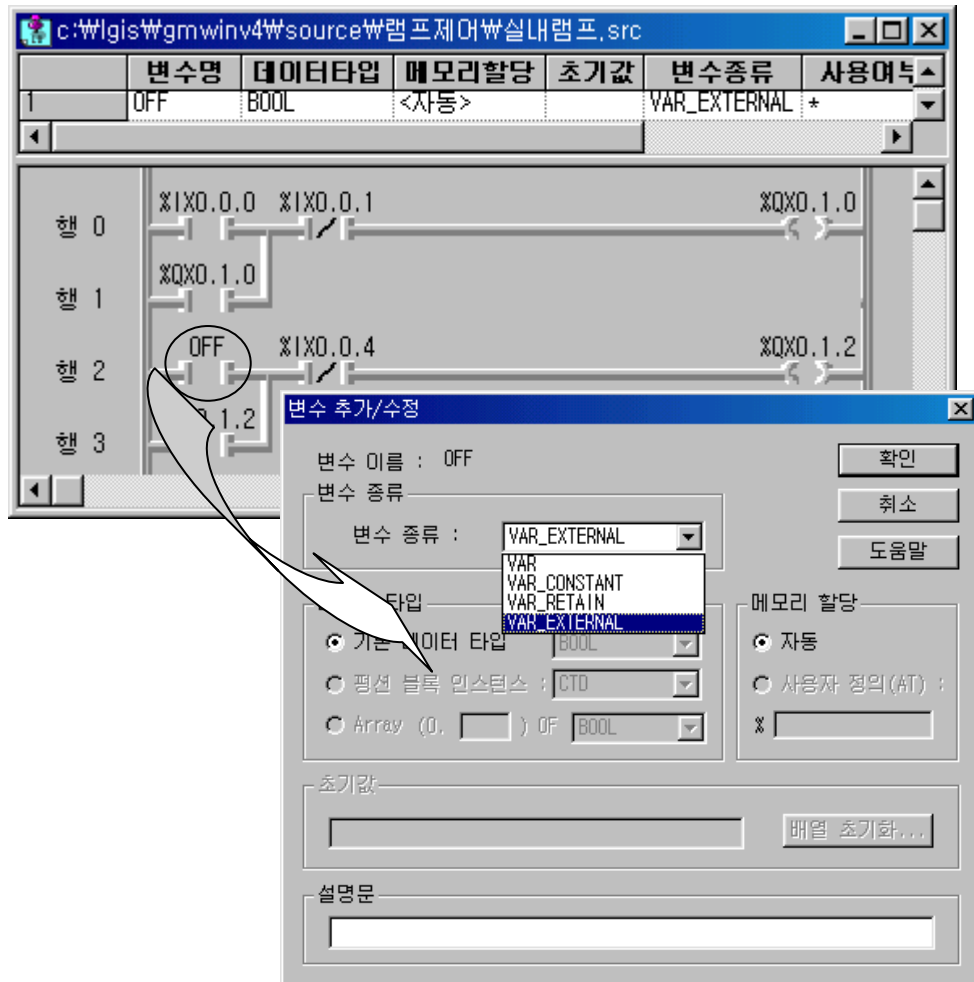
메모리 할당 : 자동

변수 종류를 VAR_EXTERNAL 로 설정합니다.

메모리 할당을 자동으로 선택합니다.

알아두기 입력 영역, 출력 영역, 내부 메모리 중 사용자 정의 메모리 할당된 영역은 글로벌 변수 등록 과정 없이 프로젝트 내의 모든 프로그램에서 사용자 정의 메모리 할당을 하면 공통으로 사용됩니다.

- ◆ 최초 VAR_EXTERNAL 로 변수가 등록된 프로그램 이 외의 프로그램에서 이 변수를 사용하기 위해서는 동일한 변수 이름을 사용하고 새로운 프로그램에서 변수의 종류를 VAR_EXTERNAL 로 선언하면 여러 개의 프로그램에서 동일한 변수 이름과 동일한 데이터를 가지고 운전할 수 있습니다.



4.3.2 파라미터

4.3.2.1 기본 파라미터

리스타트 모드는 전원을 재투입 하거나 또는 모드 전환에 의해서 RUN 모드로 운전을 시작할 때 변수 및 시스템을 어떻게 초기화한 후 RUN 모드 운전을 할 것인가를 설정하는 것으로 콜드, 웜, 핫 리스타트의 3종류가 있으며 각 리스타트 모드의 수행 조건은 다음과 같습니다.

1) 콜드 리스타트 (Cold Restart)

- 파라미터의 리스타트 모드를 콜드 리스타트로 설정 하는 경우 수행됩니다.
- 초기값이 설정된 변수를 제외한 모든 데이터를 '0'으로 리셋하고 수행합니다.
- 파라미터를 웜 리스타트 모드로 설정해도 수행할 프로그램이 변경된 후 최초 수행 시는 콜드 리스타트 모드로 수행됩니다.
- 운전 중 수동 리셋 스위치를 누르면(GMWIN에서 리셋 명령을 한 경우와 동일) 설정된 리스타트 모드에 관계없이 콜드 리스타트 모드로 수행됩니다.

2) 웜 리스타트 (Warm Restart)

- 파라미터의 리스타트 모드를 웜 리스타트로 설정 하는 경우 수행됩니다.
- 이전값 유지를 설정한 데이터는 이전값을 그대로 유지하고 초기값이 설정된 데이터는 초기값으로 설정합니다. 이외의 데이터는 '0'으로 리셋됩니다.
- 파라미터를 웜 리스타트 모드로 설정해도, 데이터 내용이 비정상일 경우(데이터의 정

전 유지가 되지 못함)에는 콜드 리스타트 모드로 수행됩니다.

- ☞ 변수의 종류를 정전 시 값이 유지되는 VAR_RETAIN으로 설정하였을 경우 다음의 규칙에 따릅니다.
- ☞ 파라미터를 워م 리스타트 모드로 설정해야 정전 시 그 값이 유지됩니다.
- ☞ 파라미터를 콜드 리스타트 모드로 설정하면 사용자가 정의한 초기값이나 기본 초기값으로 초기화됩니다.
- ☞ VAR_RETAIN으로 선언하지 않은 변수는 콜드 리스타트나 워م 리스타트 어느 경우에 도 사용자가 정의한 초기값이나 기본 초기값으로 초기화됩니다.

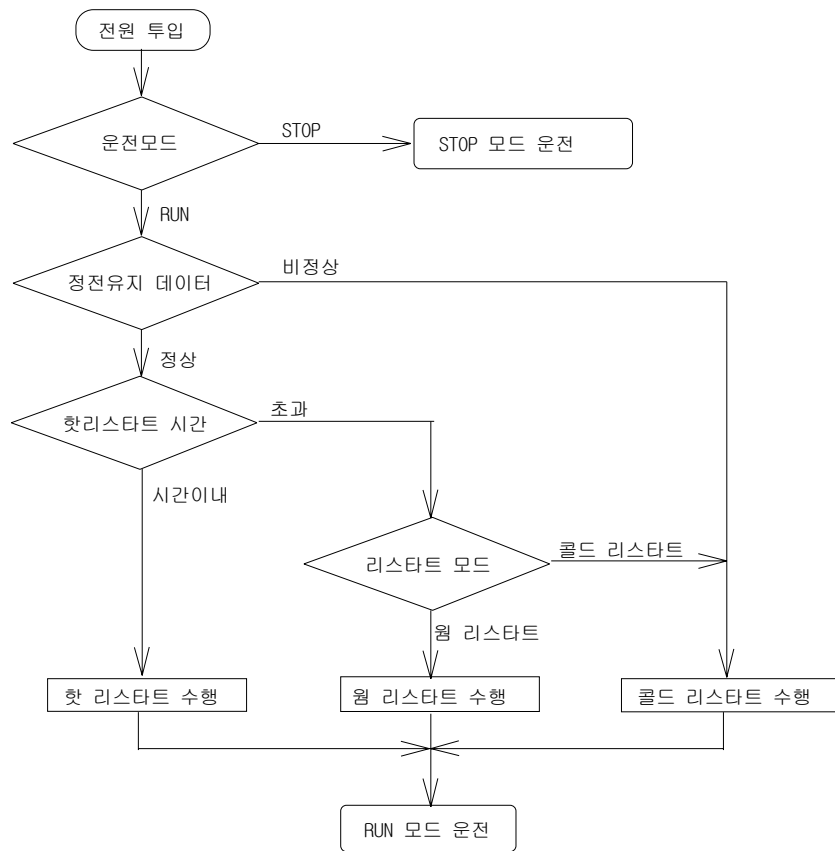
3) 핫 리스타트 (Hot Restart)

- 정상 운전 중 전원이 꺼진 후 전원이 재 투입될 때 RUN 모드이고 전원이 꺼진 후 재투입 되기 까지의 시간이 핫 리스타트 허용 시간 이내면 핫 리스타트 모드를 수행합니다.
- 모든 데이터와 프로그램 수행 요소들을 전원이 꺼지기 이전의 상태로 복원하여 수행합니다.
- 전원이 꺼지기 직전의 상태에서 다시 프로그램을 수행하므로 순간적인 정전 등에도 프로그램의 연속성을 유지할 수 있습니다.
- 핫 리스타트 허용 시간 초과 시는 파라미터에 설정된 리스타트 모드(콜드/웜)로 수행됩니다.
- 데이터 내용이 비정상일 경우(데이터의 정전 유지가 되지 못함)에는 콜드 리스타트 모드로 수행됩니다.

4) 리스타트 모드에 따른 데이터의 초기화

리스타트 모드 수행 시 각 변수에 대한 초기화 방법은 다음과 같습니다.

변수지정 모드	콜드(COLD)	웜(WARM)	핫(HOT)
디폴트	"0" 으로 초기화	"0"으로 초기화	이전값 유지
리테인	"0" 으로 초기화	이전값 유지	이전값 유지
초기화	사용자 지정값	사용자 지정값	이전값 유지
리테인 & 초기화	사용자 지정값	이전값 유지	이전값 유지



(운전 중 전원 재 투입 시 리스타트 모드 수행도)

5) 모드 변경 시 처리

- 처음 시작 시 데이터 영역의 초기화를 수행합니다.
 - ① 전원 투입시 RUN 모드일 때 → 설정된 리스타트 모드에 따름 (콜드, 웜, 핫)
 - ② ② STOP → RUN 으로 모드가 바뀔 때 → 설정된 리스타트 모드에 따름 (콜드, 웜)
- 프로그램의 유효성을 검사하여 수행 가능 여부를 판단합니다.

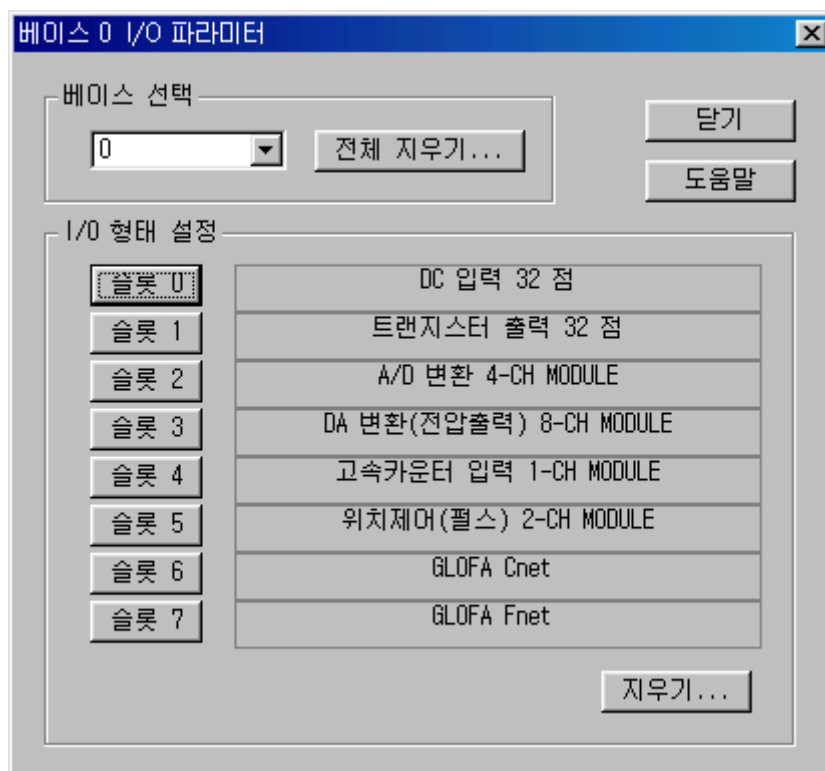
4.3.2.2 I/O 파라미터

I/O 파라미터는 PLC 가 연산을 수행하기 전에 각 슬롯에 장착되어 있는 모듈의 정보를 CPU 에 알려주는 파라미터 입니다.

I/O 파라미터를 설정하지 않으면 디폴트로 설정이 됩니다.

그러나 GMR/1/2 기종에서 I/O 파라미터를 설정하지 않으면 “모듈 불일치 에러”라는 에러가 발생합니다.

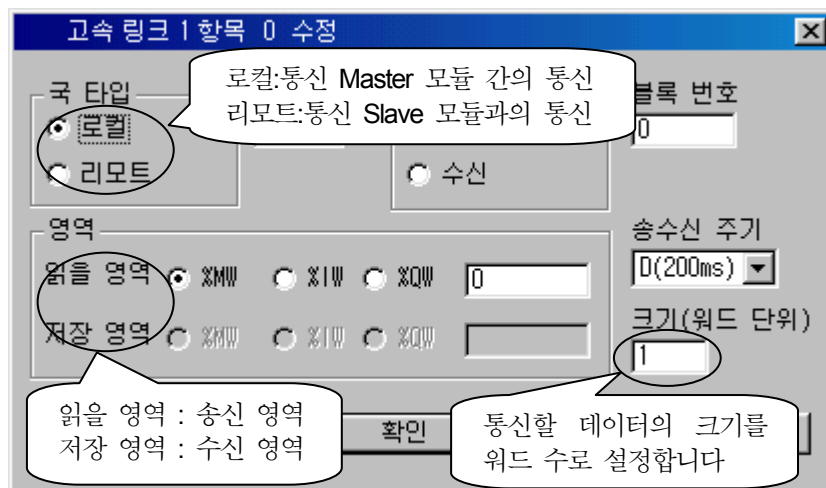
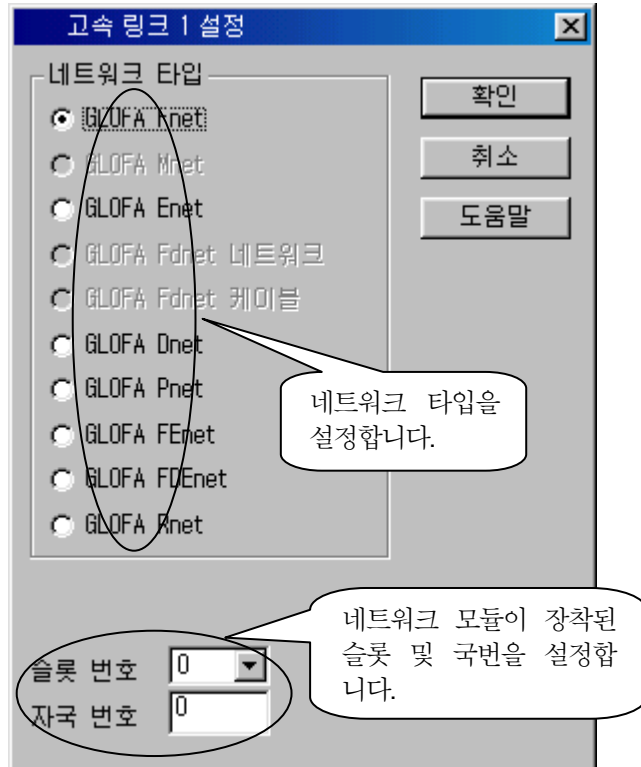
I/O 파라미터를 설정하는 방법은 구성된 시스템을 확인해서 각 슬롯별로 설정할 수가 있습니다. 그리고 PLC 와 GMWIN 이 접속된 상태에서 **온라인 메뉴 >> I/O 설정 >> I/O 정보**에서 I/O 정보를 읽어 온 후 “I/O 파라미터에 쓰기” 단추를 클릭하면 I/O 파라미터를 자동으로 설정할 수 있습니다.



4.3.3.3 고속링크 파라미터

GLOFA-GM PLC 는 프로그램에서 통신과 관련된 프로그램은 작성하지 않고 프로젝트에서 통신 파라미터만 설정해 주면 파라미터에서 설정된 주기마다 통신을 수행하는 고속 링크 통신을 지원하고 있습니다. 고속 링크 통신은 고속 링크 통신 모듈을 사용할 때에만 가능합니다.

고속 링크 통신의 종류는 Fnet, Rnet, Ethernet(Enet), Profibus-DP(Pnet), DeviceNet 이 있습니다.



알아두기

통신 Master 모듈이란 통신 명령어(데이터 읽기, 데이터 쓰기)를 실행하는 통신 모듈이며, 통신 Slave 모듈이란 Master 모듈의 요청에 의해 데이터를 송신 또는 수신할 수 있는 모듈입니다.

4.3.3 라이브러리

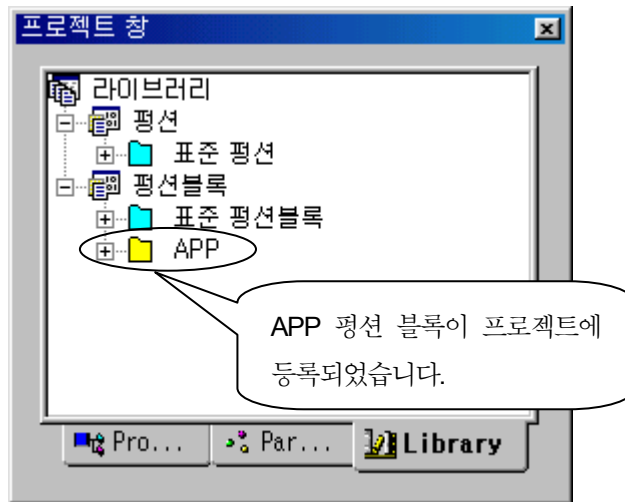
4.3.3.1 라이브러리 삽입

GLOFA-GM 시리즈 PLC 는 기종에 따라 다양한 특수 모듈과 통신 모듈을 갖추고 있습니다. 특수 모듈, 통신 모듈을 사용하기위한 평선 블록이나 특수한 명령어들(평선, 평선 블록)을 사용하기 위해서는 프로젝트에 라이브러리를 삽입해 주어야 합니다.



라이브러리 파일 이름	용도
STDLIB.xFU	기본 평선 라이브러리
STDLIB.xFB	기본 평선 블록 라이브러리
MKSTDLIB.xFU	MASTER-K 평선 라이브러리
APP.xFU	응용 평선 라이브러리
APP.xFB	응용 평선 블록 라이브러리
SPECIAL.xFB	특수 모듈을 사용하기위한 라이브러리
COMMUNI.xFB	통신 모듈을 사용하기위한 라이브러리
REMOTEn.xFB	GxL-FUEA 와 GnL-RBEA 를 이용한 리모트 제어용

프로젝트 창에서 APP 평선 블록 라이브러리가 삽입된 것을 확인할 수 있습니다.



제 5 장 프로그래밍





5.1 시퀀스 프로그램

5.1.1 시퀀스 연산자

GLOFA-GM 시퀀스 연산자는 접점(Contact), 코일(Coil), 점프(Jump) 등이 있습니다.







5.1.1.1 입력 접점의 종류 및 기능

GLOFA-GM 입력 접점의 종류 및 기능은 다음과 같습니다.

접점	이름	기능
	평상시 열린 접점	BOOL 변수의 상태가 ON 일 때에 해당 접점 연결
	평상시 닫힌 접점	BOOL 변수의 상태가 OFF 일 때에 해당 접점 연결
	양 변환 검출 접점	BOOL 변수의 값이 OFF→ON 으로 변화하는 순간 해당 접점을 현재 1 스캔 동안 연결
	음 변환 검출 접점	BOOL 변수의 값이 ON→OFF 으로 변화하는 순간 해당 접점을 현재 1 스캔 동안 연결

5.1.1.2 출력 코일의 종류 및 기능

GLOFA-GM 출력 코일의 종류 및 기능은 다음과 같습니다.

코일	이름	기능
	출력 코일	왼쪽에 있는 연결선의 상태를 지정된 BOOL 변수 접점으로 출력
	반전 코일	왼쪽에 있는 연결선의 상태를 반전하여 지정된 BOOL 변수 접점으로 출력
	양 변환 검출 코일	왼쪽 연결선 상태가 전 스캔 Off 에서 현재 스캔 On 시, 지정된 BOOL 변수 출력 접점을 현재 1 스캔 동안 ON
	음 변환 검출 코일	왼쪽 연결선 상태가 전 스캔 On 에서 현재 스캔 Off 시, 지정된 BOOL 변수 출력 접점을 현재 1 스캔 동안 ON
	셋 코일	왼쪽 연결선 상태가 1 회 ON 되었다가 OFF 되어도, 지정된 BOOL 변수 출력 접점은 ON 상태를 유지
	리셋 코일	왼쪽 연결선 상태가 ON 시, 지정된 BOOL 변수 출력 접점의 On 상태를 Off(리셋) 시킴

5.1.1.3 기타 시퀀스 연산자

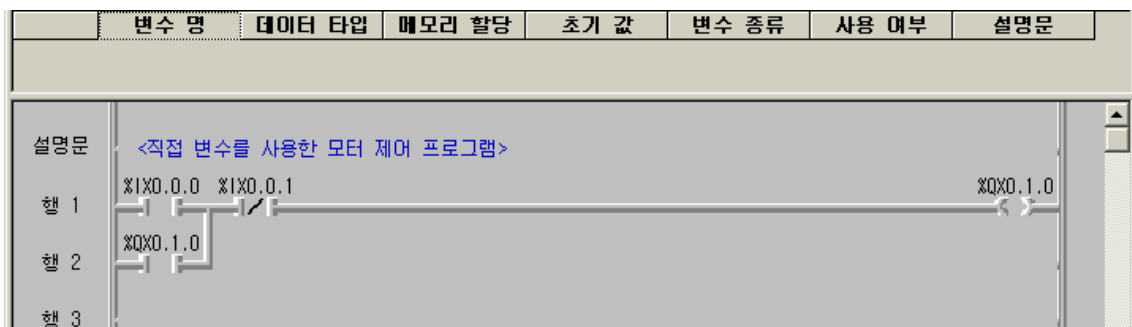
연산자	이름	기능
<SC>	서브루틴 콜	메인 프로그램 연산 도중 서브루틴 프로그램 호출
<RET>	리턴(Return)	서브루틴 연산 완료 후 메인 프로그램으로 복귀
>>	점프(Jump)	레이블 위치로 연산 이동

5.1.2 입력 접점 및 출력 코일 프로그램

5.1.2.1 직접 변수 프로그램

직접 변수를 사용하여 MASTER-K 등과 같은 방식으로 모터 제어 프로그램을 작성한 예입니다.

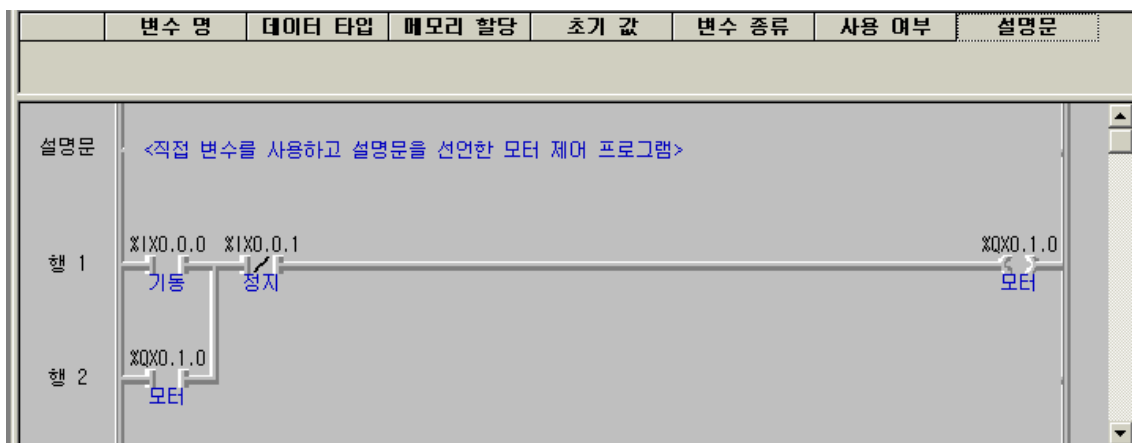
(1) 직접 변수를 사용한 모터 제어 프로그램



☞ 직접 변수를 사용하면 변수 선언이 불필요 하므로 지역 변수 목록에 포함되지 않습니다.

(2) 직접 변수를 사용하고 설명문 단 모터제어 프로그램

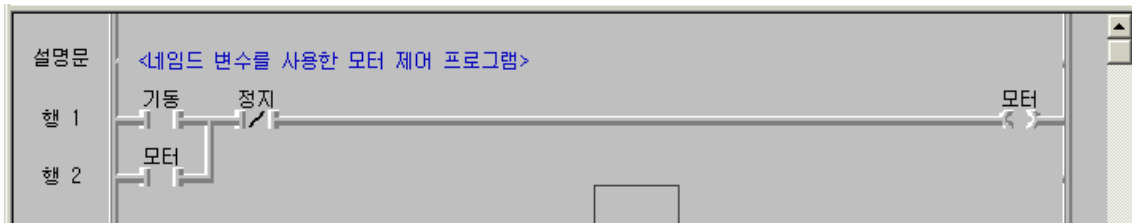
GLOFA-GM 은 직접 변수를 사용했을 경우 설명문(코멘트)을 달 수 있습니다.



5.1.2.2 네임드 변수 프로그램

네임드 변수를 사용하여 모터 제어 프로그램을 작성한 예입니다.

(1) 네임드 변수를 사용한 모터제어 프로그램



The dialog box '지역 변수 목록' (Local Variable List) displays a table of variables. A callout line points from the '기동' variable in the table to a text box below.

변수명	변수 종류	메모리 할당	사용...	데이터 타입	초기값	설명문
기동	VAR	%IX0.0.0		BOOL		푸시버튼 스위치1
정지	VAR	%IX0.0.1	*	BOOL		푸시버튼 스위치2
모터	VAR	%QX0.1.0		BOOL		콘베이어 구동용

변수명을 지정하고 메모리 할당을 사용자 정의로 한 경우
사용자 정의 메모리 할당은 직접변수 선언 방법과 동일한
방법으로 표현합니다.

(2) 네임드 변수를 사용하고 설명문을 단 모터 제어 프로그램

네임드 변수로 변수 선언을 하고 추가로 설명문(코멘트)을 달 수 있습니다.

	변수 명	데이터 타입	메모리 할당	초기 값	변수 종류	사용 여부	설명문
1	기동	BOOL	%IX0.0.0		VAR		푸시버튼 스위치1
2	정지	BOOL	%IX0.0.1		VAR	*	푸시버튼 스위치2
3	모터	BOOL	%QX0.1.0		VAR		콘베이어 구동용

설명문

<네임드 변수를 사용하고 설명문을 단 모터 제어 프로그램>

행 1

기동

정지

모터

행 2

기동

정지

모터

변수명	변수 종류	메모리 할당	사용 ...	데이터 타입	초기 값	설명문
기동	VAR	%IX0.0.0		BOOL		푸시버튼 스위치1
정지	VAR	%IX0.0.1	*	BOOL		푸시버튼 스위치2
모터	VAR	%QX0.1.0		BOOL		콘베이어 구동용

변수 추가/수정

변수 이름 : 모터

확인

변수 종류

변수 종류 : VAR

취소

도움말

데이터 타입

☒ 기본 데이터 타입

BOOL

☐ 평선 블록 인스턴스 : CTD

☐ Array (0,) of

BOOL

메모리 할당

☐ 자동

☒ 사용자 정의(AT) :

% QX0.1.0

초기 값

배열 초기화...

설명문

콘베이어 구동용

추가(A)...

수정(E)...

삭제(D)

도움말

5.1.3 변환 검출 접점, 코일 프로그램

5.1.3.1 양 변환 검출 접점 및 음 변환 검출 접점 프로그램

	변수 명	데이터 타입	메모리 할당	초기 값	변수 종류	사용 여부	설명문
1	누름_검출	BOOL	%IX0.0.0		VAR		
2	소등_스위치1	BOOL	%IX0.0.1		VAR		
3	복귀_검출	BOOL	%IX0.0.2		VAR		
4	소등_스위치2	BOOL	%IX0.0.3		VAR		
5	램프1	BOOL	%QX0.1.0		VAR		
6	램프2	BOOL	%QX0.1.1		VAR		

설명문

* 누름_검출이 0 에서 1 로 변환하는 순간을 검출하여 오른쪽 연결선이 해당 스캔에서 1 스캔 동안 연결 됩니다

설명문

* 누름_검출에 병렬 접속된 램프1을 자기 유지 회로라고 합니다

설명문

* 점등된 램프1은 소등 스위치1에 의해서 소등 됩니다

행 3

누름_검출

소등_스위치1

램프1

행 7

복귀_검출

소등_스위치2

램프2

5.1.3.2 양 변환 검출 코일 및 음 변환 검출 코일 프로그램

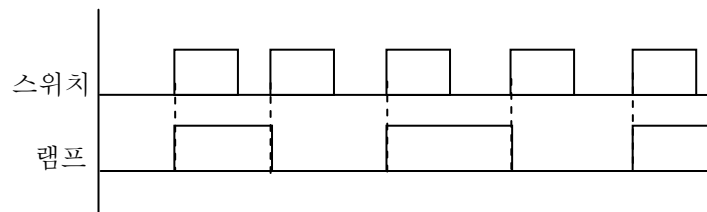
	변수 명	데이터 타입	메모리 할당	초기 값	변수 종류	사용 여부	설명문
1	기동_스위치1	BOOL	X1X0.0.0		VAR		
2	정지_스위치1	BOOL	X1X0.0.1		VAR		
3	기동_스위치2	BOOL	X1X0.0.2		VAR		
4	정지_스위치2	BOOL	X1X0.0.3		VAR		
5	모터1	BOOL	XQX0.1.0		VAR		
6	모터2	BOOL	XQX0.1.1		VAR		
7	누름_검출	BOOL	<자동>		VAR		
8	복귀_검출	BOOL	<자동>		VAR		

설명문	* 기동 스위치1을 누르는 순간 모터1은 기동되고, 정지 스위치1에 의해서 정지 됩니다						
행 1	기동 스위치1						
행 2	정지 스위치1					누름_검출	
행 3	모터1					모터1	
행 4	정지 스위치1						
설명문	* 기동 스위치2를 눌렀다 떼는 순간 모터2는 기동됩니다.						
행 6	기동 스위치2						
행 7	복귀_검출					복귀_검출	
행 8	모터2					모터2	
행 9	정지 스위치2						

♣ 다이내믹 플립프롭

스위치를 한 번 OFF →ON 하면 램프가 ON 되고 다시 OFF →ON 하면 램프가 OFF 됩니다.

♣ 타임차트



♣ 프로그램 및 변수 설정

c:\wlgis\wgmwin\source\플립플롭\wnoname00.src

	변수명	데이터타입	메모리할당	초기값	변수종류	사용여부	설명문
1	램프	BOOL	%QX0.1.0		VAR	*	LED
2	스위치	BOOL	%IX0.0.0		VAR	*	신호 입력
3	펄스	BOOL	<자동>		VAR	*	펄스

행	스위치	펄스	램프
행 0	스위치	펄스	램프
행 1	스위치	펄스	램프
행 2	스위치	펄스	램프

♣ 모터의 기동 수 제어

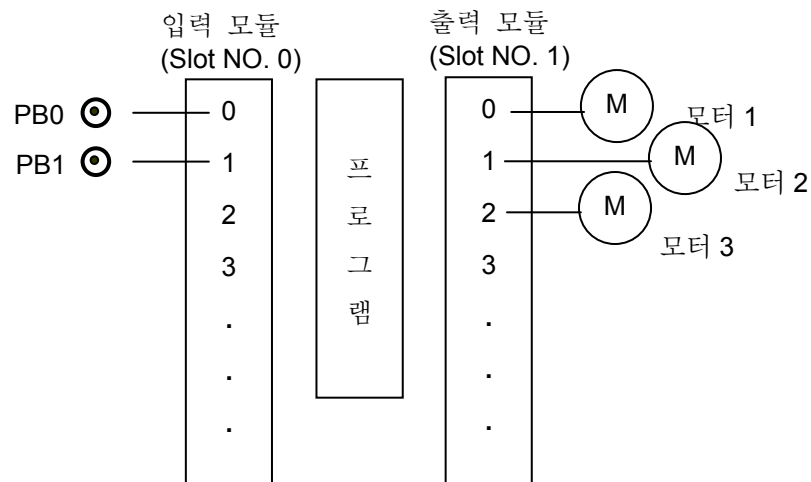
순간 접촉 푸쉬 버튼 PB0 을 첫번째 누르면 모터 1 이 ON, 두번째 누르면 모터 2 가 ON, 세번째 누르면 모터 3 이 ON 됩니다. 결국 순간 접촉 푸쉬 버튼 PB0 를 세 번 누르면 세대의 모터가 모두 기동하게 됩니다.

순간 접촉 푸쉬 버튼 PB1 을 누르면 모든 모터의 기동이 중지됩니다.

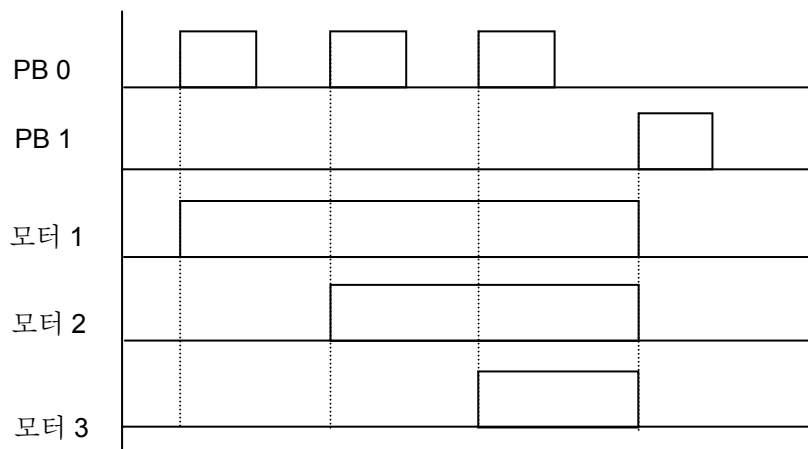
본 예제는 PLC 시퀀스 의 정확한 이해로 작성할 수 있습니다.

직렬처리방식과 입출력 리프레시의 관계를 고려하여야만 정확한 결과를 나타낼 수 있습니다.

♣ 시스템도



♣ 타임차트

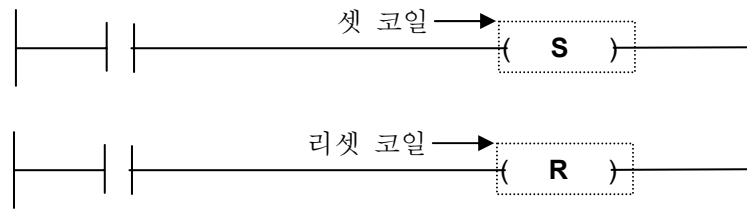


♣ 프로그램 및 변수 설정

c:\wlgis\wgmwin\source\모터제어\noname00.src

	변수명	데이터타입	메모리할당	초기값	변수종류	사용여부	설명문
1	모터1	BOOL	%QX0.0.0		VAR	*	
2	모터2	BOOL	%QX0.1.1		VAR	*	
3	모터3	BOOL	%QX0.1.2		VAR	*	
4	정지	BOOL	<자동>		VAR	*	
5	펄스	BOOL	<자동>		VAR	*	
6	PB0	BOOL	%IX0.0.0		VAR	*	
7	PB1	BOOL	%IX0.0.1		VAR	*	

5.1.4 셋 및 리셋 코일 프로그램



구 분	사용 가능 영역
S	%Q, %M
R	%Q, %M

◎ 셋 코일

입력 조건이 ON 되면 지정된 비트 영역이 ON 됩니다. 지정된 비트 영역이 ON 된 후 입력 조건이 OFF 되어도 지정된 비트 영역은 ON 상태를 유지합니다.

◎ 리셋 코일

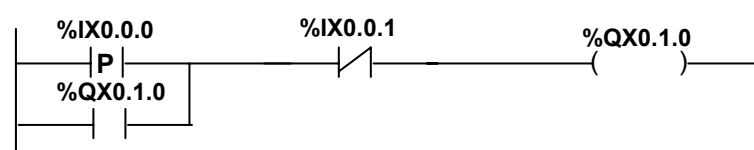
리셋 코일의 입력 조건이 ON 되었을 때, 리셋 코일로 지정된 비트 영역이 ON 상태이면 OFF 상태로 만듭니다.

리셋 코일의 입력 조건이 ON 되었을 때, 리셋으로 지정된 비트 영역이 OFF 상태이면 아무런 변화도 일어나지 않습니다.

◎ 참고 사항 - 자기 유지 회로

푸시 버튼 스위치 입력 조건을 받아 출력 코일을 ON 시킨 후 푸시 버튼에서 손을 떼어도 계속 출력을 ON 으로 유지 시킬 수 있는 방법은 자기 유지 회로 방식과 SET/RESET 코일을 이용하는 두 가지 방법이 있습니다.

자기 유지 회로란 출력 접점을 한 번 작동시킨 후 그 출력 접점의 ON/OFF 정보를 다시 자기 입력으로 받는 방식입니다.



■ 셋 및 리셋 프로그램 예

	변수 명	데이터 타입	메모리 할당	초기 값	변수 종류	사용 여부	설명문
1	투입SW	BOOL	%IX0.0.0		VAR		
2	차단SW	BOOL	%IX0.0.1		VAR		
3	램프	BOOL	%QX0.1.0		VAR		
4	래치	BOOL	<자동>		VAR RETAIN	*	

설명문	* 셋/리셋 코일은 자기유지 기능을 갖고 있기 때문에 코일 출력이 1회 셋 되면 '차단' 입력이 들어올 때까지 그 상태를 유지합니다.						
설명문	* 정전 후 복전 시 변수 종류가 VAR인 '램프'와 VAR_RETAIN인 '래치'는 동작이 다릅니다.						
설명문	* 리스타트 모드가 월 리스타트로 설정되고, VAR_RETAIN으로 선언된 변수 '래치'는 정전 후 복전 시 데이터 상태는 보존됩니다.						
행 3							
행 4	투입SW						램프 (S)
행 5							래치 (S)
행 6							
행 7	차단SW						램프 (R)
행 8							래치 (R)

기본 파라미터

컨피그레이션 (PLC) 이름:

래치코일

적용 PLC 버전: v1.7

☒ 통신에 의한 PLC 제어 허용

☐ 핫 리스타트

시간

분

초

리스타트 모드

☐ 콜드 리스타트

☒ 월 리스타트

리소스 (CPU) 설정

리소스 0

이름

타입

스캔 위치독

GM4

200 ms

확인

취소

도움말

5.2 평선 프로그램

5.2.1 평선(Function)과 평선 블록(Function Block)

GLOFA-GM 시리즈 PLC 에서 사용되는 언어 구성체는 크게 평선과 평선 블록으로 구분됩니다.

- ▷ 평선은 입력에 대한 연산 결과를 1 스캔에 즉시 출력합니다.
- ▷ 평선은 출력이 하나입니다.

평선은 1 스캔에 입력을 받아 동일 스캔에 연산을 실행하여 그 결과를 만들어 내는 언어 구성체입니다.

- ▷ 평선 블록은 여러 스캔에 걸쳐 누계된 연산 결과를 출력합니다.
- ▷ 평선 블록은 출력이 여러 개가 될 수 있습니다.

평선 블록은 여러 스캔에 걸쳐 누계된 연산 결과를 출력하므로, 연산 중 누계되는 데이터를 보관하기 위한 내부 메모리가 필요합니다.

따라서 평선 블록은 사용하기 전에 인스턴스 변수를 선언 합니다.

인스턴스 변수는 평선 블록 내에서 사용하는 변수들의 집합입니다.

평선과 평선 블록은 차이점을 표로 나타내면 다음과 같습니다.

구분	평선	평선 블록
입력의 수	1 개 이상(최대 8 개)	2 개 이상
출력의 수	오직 1 개	1 개 이상
연산 시간	1 스캔에 결과 출력	여러 스캔 누계 결과 출력
데이터	입, 출력 데이터를 모두 반드시 지정	입력 데이터는 반드시 지정하고, 출력 데이터는 생략 가능함
데이터 타입	입력 변수와 출력 변수의 모든 데이터 타입이 동일	변수의 기능에 따라 다양한 데이터 타입
예	전송 평선, 형 변환 평선, 비교 평선, 산술 연산 평선 등	타이머, 카운터, 응용 평선 블록 특수모듈 초기화 평선 블록 등

알아 두기 평선 또는 평선 블록 사용 시 여러 개의 입, 출력 변수가 있어도 좌측 모션 과 우측 모션에 연결한 수 있는 입력 단자는 1 개이며, 우측 모션에 연결할 수 있는 출력 단자도 1 개 입니다.

5.2.2 기본 평선의 종류

기본 평선에는 전송 평선, 형 변환 평선, 비교 평선, 산술 연산 평선, 논리 연산 평선, 비트 시프트 평선 등이 있습니다.

5.2.2.1 전송 평선

평선 이름	기 능
MOVE	데이터 전송 (IN →OUT)
ARY_MOVE	배열 변수 부분 전송

5.2.2.2 비교 평선

비교 결과가 참(True)이면 OUT 으로 1 이 출력됩니다.

평선 이름	기 능(단, n은 8까지 가능)
GT >	‘크다’ 비교 (IN1 > IN2) And (IN2 > IN3) And ... And (INn-1 > INn) → OUT)
GE ≥	‘크거나 같다’ 비교 (IN1 ≥ IN2) And (IN2 ≥ IN3) And ... And (INn-1 ≥ INn) → OUT)
EQ =	‘같다’ 비교 (IN1 = IN2) And (IN2 = IN3) And ... And (INn-1 = INn) → OUT)
LE ≤	‘작거나 같다’ 비교 (IN1 ≤ IN2) And (IN2 ≤ IN3) And ... And (INn-1 ≤ INn) → OUT)
LT <	‘작다’ 비교 (IN1 < IN2) And (IN2 < IN3) And ... And (INn-1 < INn) → OUT)
NE ≠	‘같지 않다’ 비교 (IN1 ≠ IN2) And (IN2 ≠ IN3) And ... And (INn-1 ≠ INn) → OUT)

5.2.2.3 산술 연산 평션

산술 연산 평션 중 일반적인 것은 사칙 연산(덧셈, 뺄셈, 곱셈, 나눗셈, 나머지) 평션입니다.

평션 이름	기 능
ADD	더하기 ($IN1 + IN2 + \dots + INn \rightarrow OUT$) (단, n 은 8 까지 가능)
MUL	곱하기 ($IN1 \times IN2 \times \dots \times INn \rightarrow OUT$) (단, n 은 8 까지 가능)
SUB	빼기 ($IN1 - IN2 \rightarrow OUT$)
DIV	나누기 ($IN1 \div IN2 \rightarrow OUT$)
MOD	나눗셈 나머지 구하기

5.2.2.4 논리 연산 평션

평션 이름	기 능 (단, n은 8까지 가능)
AND	논리곱 ($IN1 \text{ AND } IN2 \text{ AND } \dots \text{ AND } INn \rightarrow OUT$)
OR	논리합 ($IN1 \text{ OR } IN2 \text{ OR } \dots \text{ OR } INn \rightarrow OUT$)
XOR	배타적 논리합 ($IN1 \text{ XOR } IN2 \text{ XOR } \dots \text{ XOR } INn \rightarrow OUT$)
NOT	논리 반전 ($\text{NOT } IN1 \rightarrow OUT$)

5.2.2.5 비트 시프트 평션

평션 이름	기 능
SHL	입력을 N 비트 왼쪽으로 이동(오른쪽은 0 으로 채움)
SHR	입력을 N 비트 오른쪽으로 이동(왼쪽은 0 으로 채움)
ROL	입력을 N 비트 왼쪽으로 회전
ROR	입력을 N 비트 오른쪽으로 회전

5.2.2.6 형(Type) 변환 평선

평선 이름	입력 데이터 형	종류	적용 기종
BCD_TO_***	BCD	BCD_TO_SINT 등 8종	전기종
INT_TO_***	INT	INT_TO_SINT 등 15종	
SINT_TO_***	SINT	SINT_TO_INT 등 15종	
DINT_TO_***	DINT	DINT_TO_SINT 등 15종	
UINT_TO_***	UINT	UINT_TO_SINT EMD 16종	
USINT_TO_***	USINT	USINT_TO_SINT 등 15종	
UDINT_TO_***	UDINT	UDINT_TO_SINT 등 17종	
WORD_TO_***	WORD	WORD_TO_SINT 등 14종	
DWORD_TO_***	DWORD	DWORD_TO_SINT 등 16종	
BOOL_TO_***	BOOL	BOOL_TO_SINT 등 13종	
BYTE_TO_***	BYTE	BYTE_TO_SINT 등 13종	
TIME_TO_***	TIME	TIME_TO_UDINT 등 3종	
DATE_TO_***	DATE	DATE_TO_UINT 등 3종	
DT_TO_***	DT	DT_TO_LWORD 등 4종	
TOD_TO_***	TOD	TOD_TO_UDINT 등 3종	
STRING_TO_***	STRING	STRING_TO_SINT 등 19종	
NUM_TO_***	NUM	NUM_TO_STRING	
LWORD_TO_***	LWORD	LWORD_TO_SINT 등 15종	GMR/1/2
LINT_TO_***	LINT	LINT_TO_SINT 등 15종	
ULINT_TO_***	ULINT	ULINT_TO_SINT 등 15종	
REAL_TO_***	REAL	REAL_TO_SINT 등 10종	
LREAL_TO_***	LREAL	LREAL_TO_SINT 등 10종	

알아 두기 형 변환 평선 중 전기종 적용 명령어라도 출력 타입이 L***, UL***는 GMR/1/2 기종에서만 사용 가능 합니다.

예) INT_TO_LWORD
WORD_TO_ULINT

5.2.3 기본 평선 프로그램

5.2.3.1 기본 평선 프로그램 작성

GMWIN 에서 평선을 편집하는 방법에 대해서 설명합니다.

- ◆ 새 프로젝트를 다음과 같이 작성합니다.


프로젝트 이름 : 전송 평선 예제

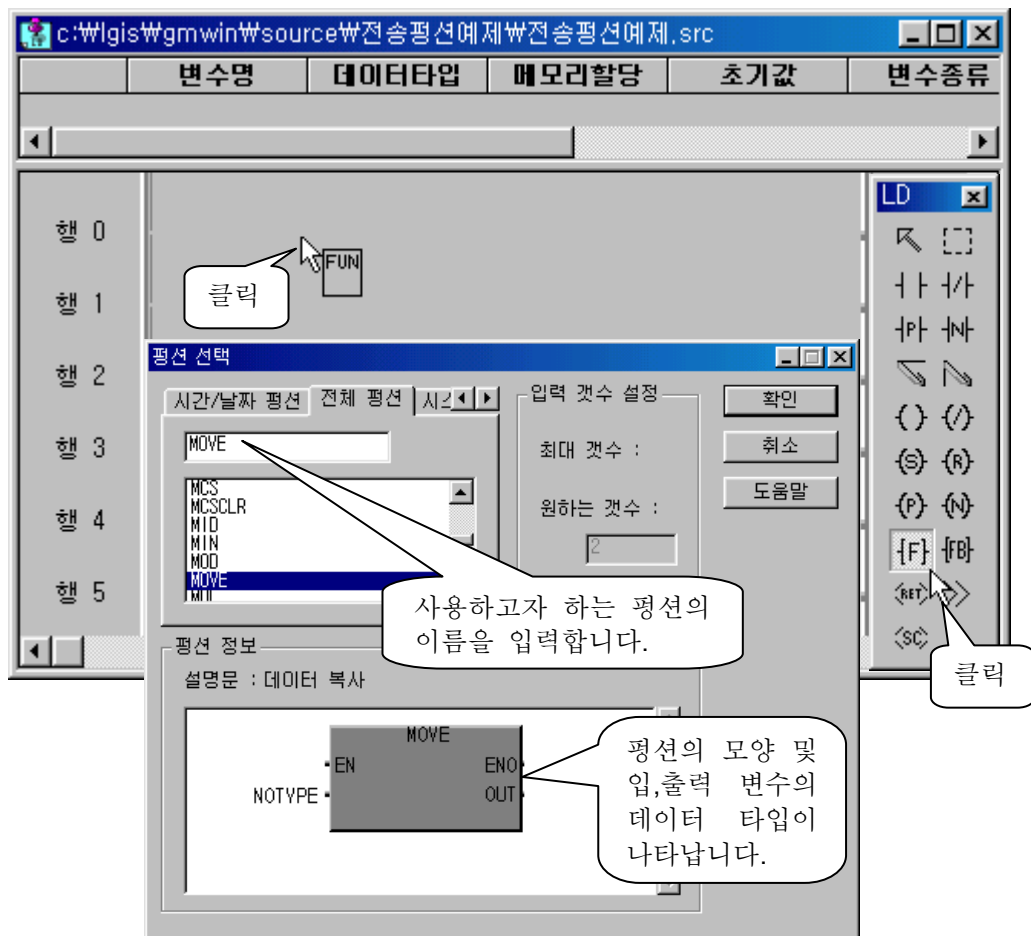
프로그램 파일 이름 : 전송 평선 예제

PLC 기종 : GM4

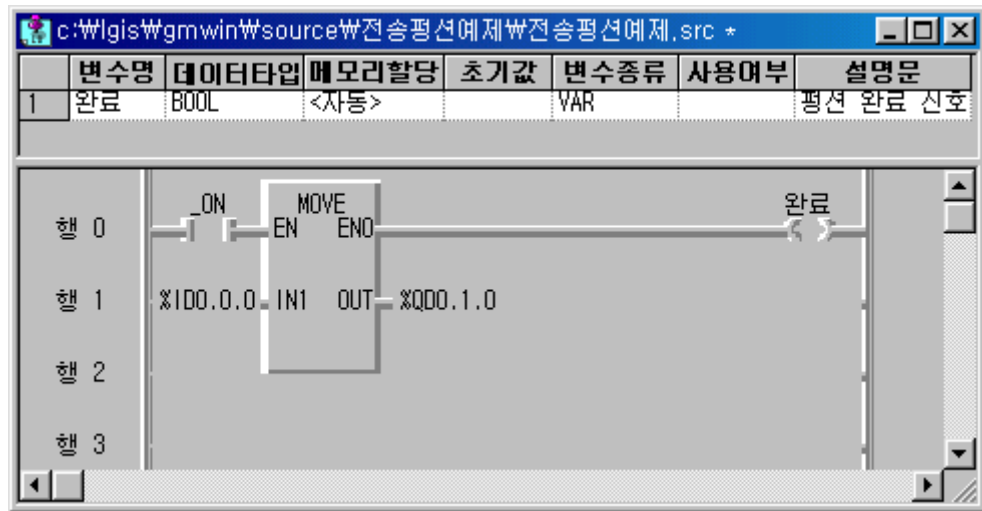
사용 언어 : LD

프로그램의 종류 : 프로그램 블록

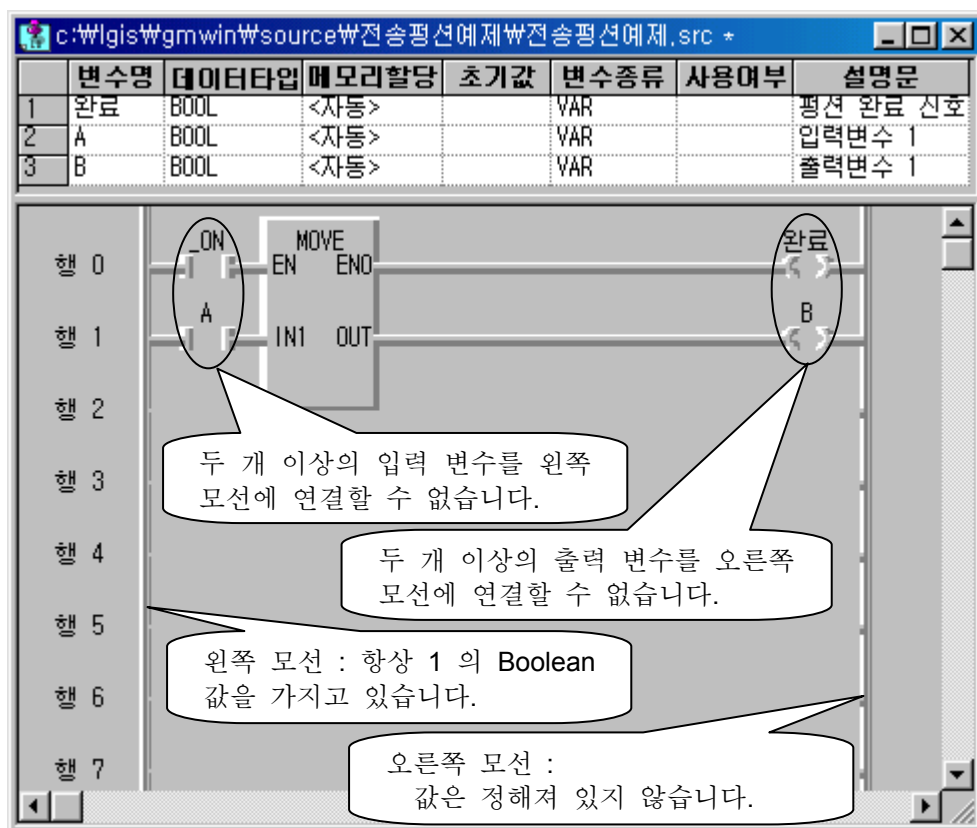
- ◆ 도구 바에서 평선()을 선택한 후 프로그램 창에서 클릭하면 평선 선택 화면이 나타납니다. 평선 선택 화면에서 사용하고자 하는 평선의 이름을 입력하고 “확인”을 클릭하면 프로그램 창에 평선이 등록됩니다.



- ◆ 평선이 등록되었으면 시퀀스 연산자를 이용하여 입력 변수 및 출력 변수를 설정합니다.

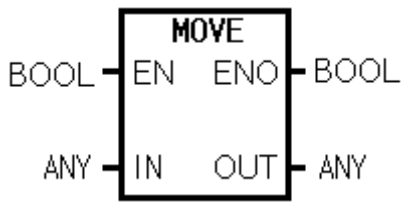


- ◆ 다음과 같이 입력 변수 2 개 이상을 좌측 모선에 연결 하거나 출력 변수 2 개 이상을 우측 모선에 연결하면 에러가 발생합니다.



5.2.3.2 전송 평선

MOVE

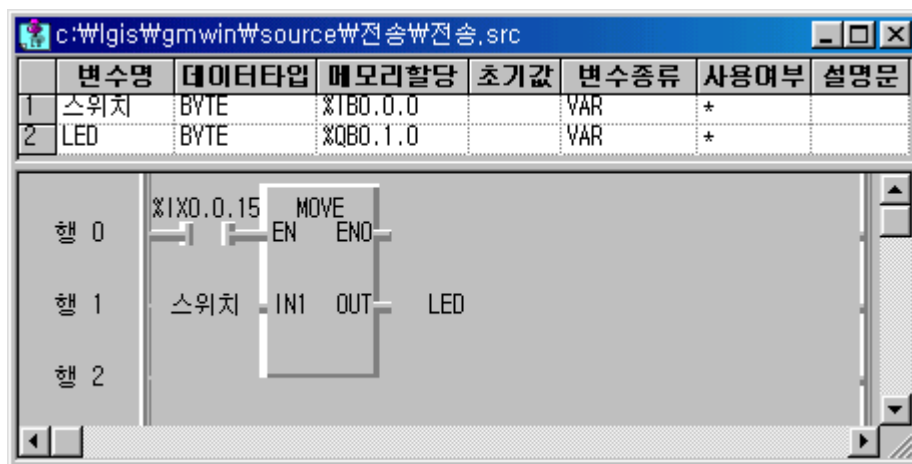
평 선	설 명
	<p>입력:</p> <p>EN: EN 이 1 일 때 평선 실행</p> <p>IN: 전송할 값 또는 전송할 데이터가 저장된 변수</p> <p>출력:</p> <p>ENO: 평선이 수행되면 1 출력</p> <p>OUT: 데이터가 저장될 영역</p>

● 기능

▷ EN 이 ON 되면 IN 으로 입력되는 데이터를 OUT 으로 전송합니다.

● 프로그램 예 1

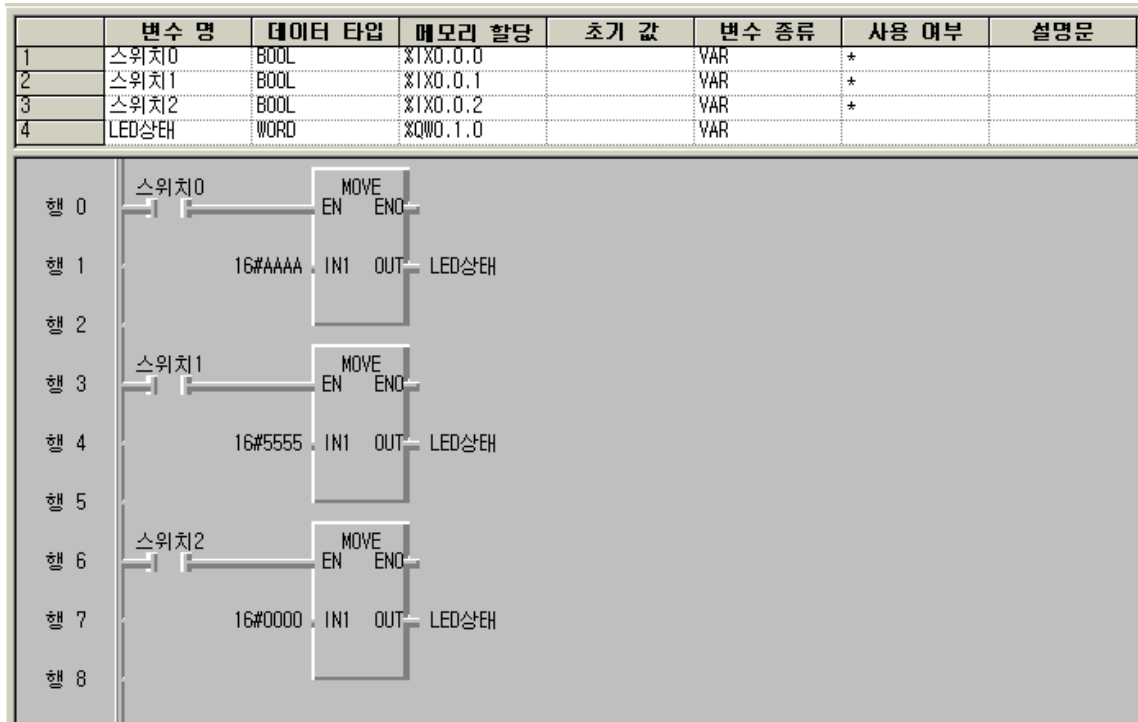
▷ 실행 조건이 ON 되면 스위치의 하위 1Byte(%IB0.0.0 ~ %IX0.0.7)까지의 ON/OFF 정보가 복사되어 LED 의 하위 1Byte(%QB0.1.0 ~ %QX0.1.7)로 전송됩니다.



알아 두기 평선에서 EN 은 입력 변수가 아닌 평선 실행의 조건이며, ENO 는 출력 변수가 아닌 평선 실행 완료 신호입니다.
그리고, 입력 변수와 출력 변수의 데이터 타입은 동일해야 합니다.

◎ 프로그램 예 2

스위치 0, 1, 2 중 하나를 ON 하면 MOVE 평선이 실행되어 해당 코드값을 LED(%QW0.1.0)로 전송합니다.



5.2.3.3 형 변환 평선

BCD_TO_***

평 선	설 명
	<p>입력:</p> <p>EN:EN 이 1 일 때 평선 실행</p> <p>IN:BCD 형태의 데이터를 갖는 ANY_BIT 입력값</p> <p>출력:</p> <p>ENO:평선이 수행되면 1 출력</p> <p>OUT:타입 변환된 데이터가 저장될 변수</p>

◎ 기능

- ▷ BCD 코드의 입력 데이터를 바이너리 코드(정수)로 바꾸어 OUT 으로 설정된 변수에 저장합니다.

평 선	입력 타입	출력 타입	입력 범위
BCD_TO_SINT	BYTE	SINT	16#00 ~ 16#99
BCD_TO_USINT	BYTE	USINT	
BCD_TO_INT	WORD	INT	16#0000~16#9999
BCD_TO_UINT	WORD	SINT	
BCD_TO_DINT	DWORD	DINT	16#00000000 ~ 16#99999999
BCD_TO_UDINT	DWORD	UDINT	

◎ 프로그램 예

- ▷ 디지털 스위치(%IW0.0.1)를 사용하여 BCD 값을 입력하고, 스위치 0 (%IX0.0.0)를 ON 하면 정수로 변환되어 정수값에 저장됩니다.
만일, 입력의 데이터가 BCD 형이 아닐 경우, 에러 램프가 ON 됩니다.

알아 두기 BCD 코드란 A~F 까지를 사용할 수 없는 16 진수를 말합니다. 따라서 입력 변수에 16#1A, 16#AF 등은 사용할 수 없습니다.
입력 변수가 BCD 형이 아닐 경우 출력은 0 이 되고, _ERR(연산 에러 플래그), _LER(연산 에러 래치 플래그)가 ON 됩니다.

c:\wlgis\gmwin\source\bcd_to_int\bcd_to_int.src *

변수명	데이터타입	메모리할당	초기값	변수종류	사용여부	설명문
1	디지털_스위치	WORD	%IWO.0.1	VAR		
2	에러	BOOL	%QX0.1.0	VAR		연산 에러 램프
3	정수값	INT	<자동>	VAR	*	

설명문

설명문

행 2

행 3

행 4

행 5

행 6

행 7

디지털 스위치(%IWO.0.1)를 사용하여 BCD 값을 입력하고, 스위치0 (%IX0.0.0)를 ON 하면 정수로 변환되어 정수값에 저장됩니다.

만일, 입력의 데이터가 BCD형이 아닐 경우, 에러 램프가 ON 됩니다.

INT_TO_***

평 선	설 명
	<p>입력:</p> <p>EN:EN 이 1 일 때 평선 실행</p> <p>IN:BCD 형태의 데이터를 갖는 ANY_BIT 입력값</p> <p>출력:</p> <p>ENO:평선이 수행되면 1 출력</p> <p>OUT:타입 변환된 데이터가 저장될 변수</p>

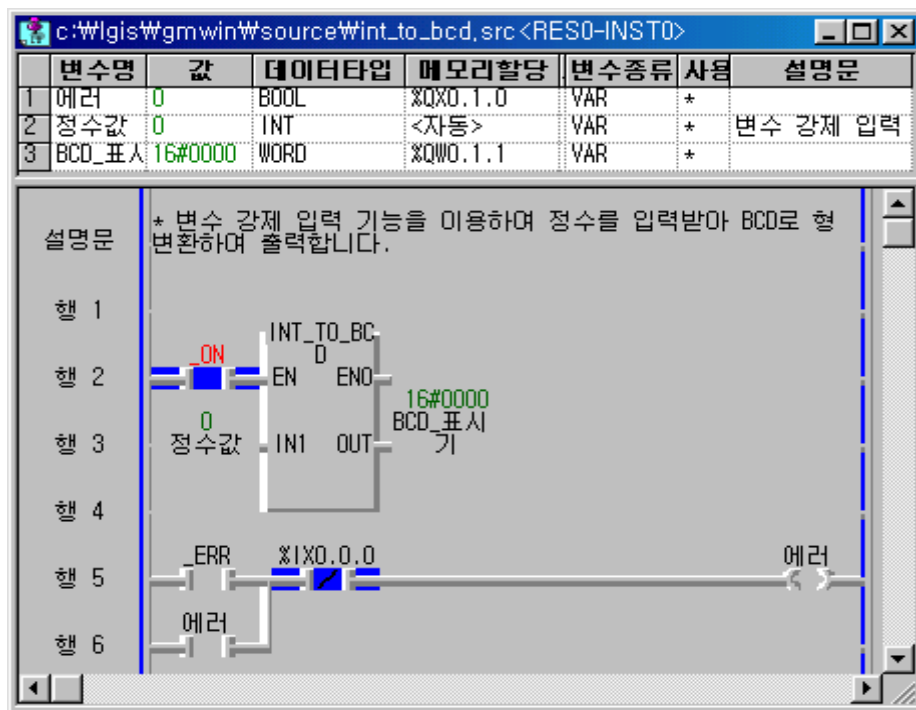
● 기능

- ▷ 정수형의 입력 데이터를 BCD 코드로 형 변환하여 OUT 으로 설정된 변수에 저장합니다.

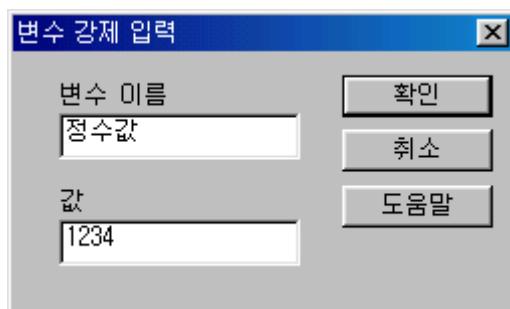
평 선	출력 타입	동작 설명
INT_TO_SINT	SINT	입력이 -128 ~ 127 일 때 정상 변환
INT_TO_USINT	USINT	입력이 0 ~ 255 일 때 정상 변환
INT_TO_UINT	UINT	입력이 0 ~ 32767 일 때 정상 변환
INT_TO_DINT	DINT	DINT 로 형 변환
INT_TO_UDINT	UDINT	입력이 0~32767 일 때 정상 변환
INT_TO_BOOL	BOOL	하위 1 비트를 취해서 BOOL 로 변환
INT_TO_BYTE	BYTE	하위 1 바이트를 취해서 BYTE 로 변환
INT_TO_WORD	WORD	비트 배열의 변화없이 WORD 로 변환
INT_TO_DWORD	DWORD	상위 비트열을 0 으로 채움
INT_TO_LWORD	LWORD	상위 비트열을 0 으로 채움
INT_TO_BCD	WORD	입력이 0 ~ 9999 일 때 정상 변환

◎ 프로그램 예

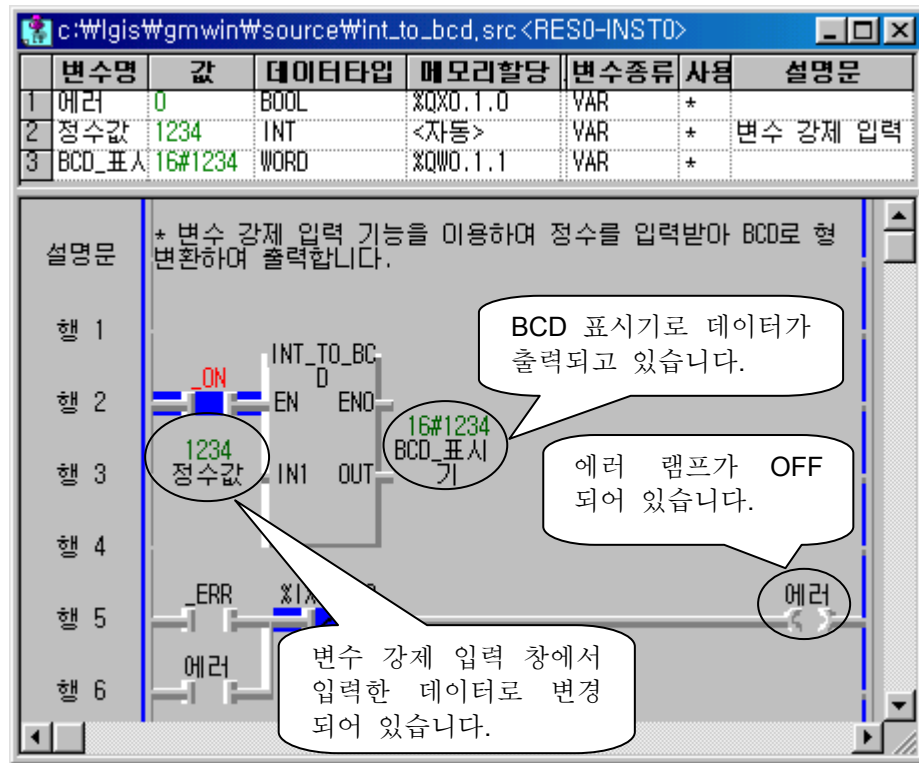
- ▷ 변수 강제 입력 기능을 이용하여 정수값에 0 ~ 9999 사이의 임의의 값을 입력했을 때 BCD 타입으로 형 변환하여 BCD 표시기로 표시됩니다. 입력값이 0 ~ 9999 사이의 값이 아닐 경우 에러 램프가 ON 됩니다.



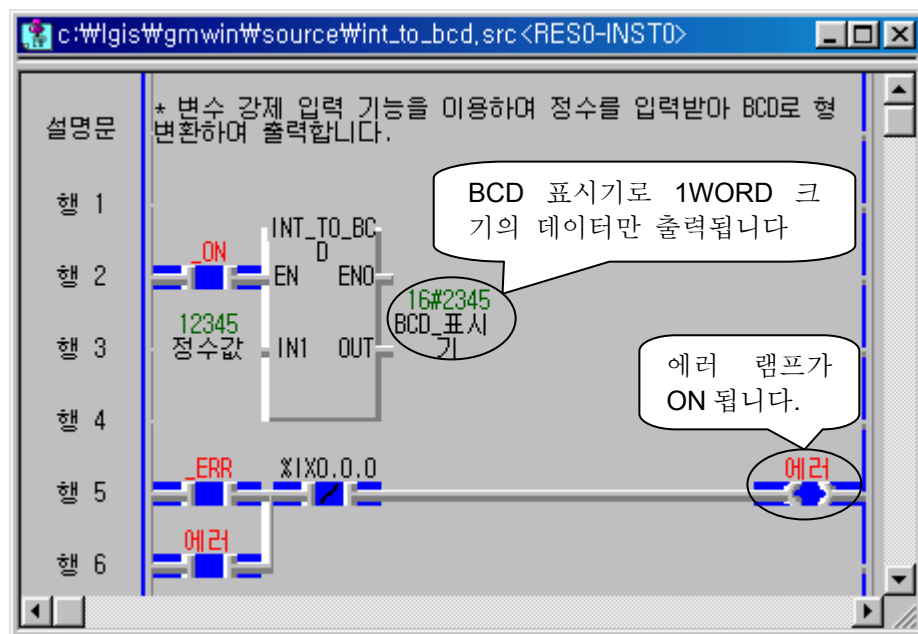
- ◆ 프로그램을 작성하여 PLC 로 전송한 후 모니터를 실행합니다. 모니터를 실행하면서 평선의 입력 변수 '정수값'을 더블 클릭하면 다음과 같은 변수 강제 입력 창이 나타납니다.



- ◆ 변수 강제 입력 창에서 0 ~ 9999 사이의 임의의 값을 입력하고 확인을 클릭합니다.



- ◆ 변수 강제 입력 창에서 0 ~ 9999 이외의 임의의 값을 입력하고 확인을 클릭합니다.



5.2.3.4 비교 평선

GT

평 선	설 명
	<p>입력:</p> <p>EN:EN 이 1 일 때 평선 실행</p> <p>IN1~IN8:비교할 데이터</p> <p>출력:</p> <p>ENO:평선이 수행되면 1 출력</p> <p>OUT:비교 결과가 저장될 영역</p> <p>IN1>IN2... >INn 을 만족하면 1 출력</p>

● 기능

입력값의 비교 결과 $IN1 > IN2 > IN3 \dots > INn$ (n 은 입력 개수, 8 까지 가능) 이 참이면 OUT 으로 1 을 출력합니다.

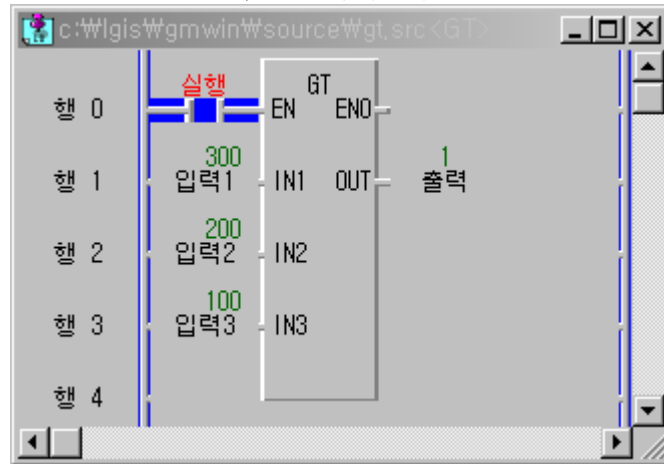
$IN1 > IN2 > IN3 \dots > INn$ 의 조건 중 하나라도 만족하지 않으면 OUT 으로 0 을 출력합니다.

● 프로그램 예

입력 1, 입력 2, 입력 3 세 개의 입력을 받아 입력 1>입력 2>입력 3 의 조건을 만족하면 출력 LED 램프가 ON 됩니다.

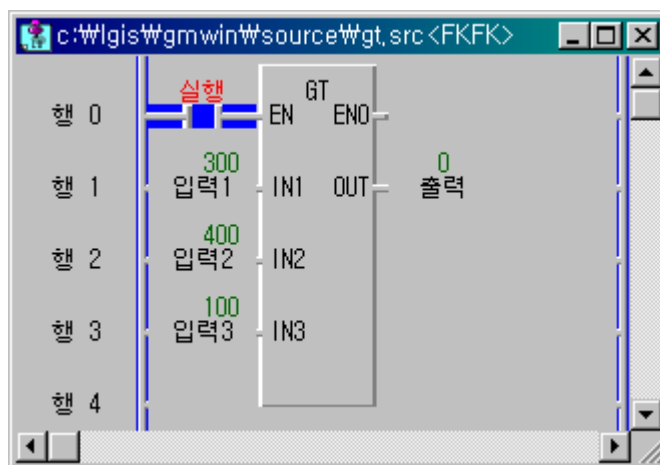
변수명	변수 종류	메모리할당	사용여부	데이터 타입
실행	VAR	%IX0.0.0	*	BOOL
입력1	VAR	<자동>	*	INT
입력2	VAR	<자동>	*	INT
입력3	VAR	<자동>	*	INT
출력	VAR	%QX0.1.0	*	BOOL

<프로그램 및 지역변수>



<모니터링 1>

입력 1>입력 2>입력 3 을 만족하므로 OUT 단자에 1 이 출력되며, LED 가 ON 됩니다.



<모니터링 2>

입력 1>입력 2>입력 3 을 만족하지 못하므로 OUT 단자에 0 이 출력되며, LED 가 OFF 됩니다.

GE

평 선	설 명
	<p>입력:</p> <p>EN:EN 이 1 일 때 평선 실행</p> <p>IN1~IN8:비교할 데이터</p> <p>출력:</p> <p>ENO:평선이 수행되면 1 출력</p> <p>OUT:비교 결과가 저장될 영역</p> <p>IN1≥IN2...≥INn 을 만족하면 1 출력</p>

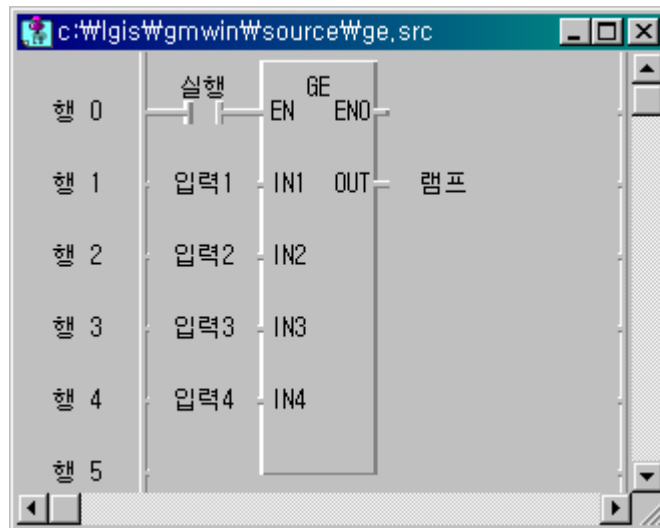
◎ 기능

입력값의 비교 결과 $IN1 \geq IN2 \geq IN3 \dots \geq INn$ (n 은 입력 개수, 8 까지 가능)을 만족하면 OUT 으로 1 을 출력합니다.

$IN1 \geq IN2 \geq IN3 \dots \geq INn$ 을 만족하지 못하는 경우에는 OUT 으로 0 을 출력합니다.

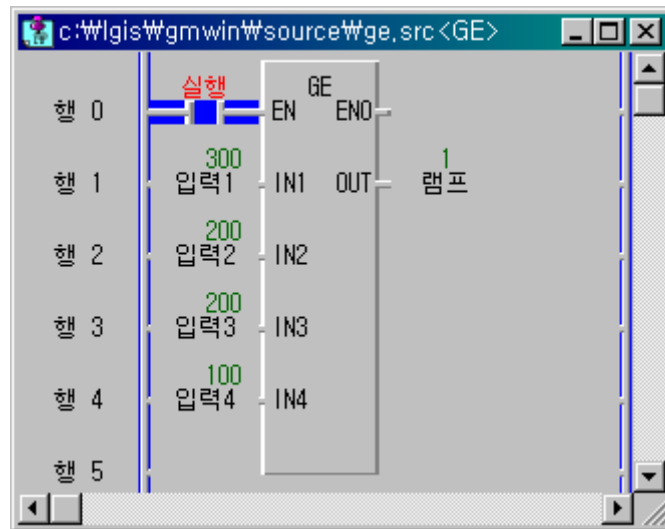
◎ 프로그램

4 개의 입력을 받아 입력 1≥입력 2≥입력 3≥입력 4 의 조건을 만족하면 출력 LED 램프가 ON 됩니다.



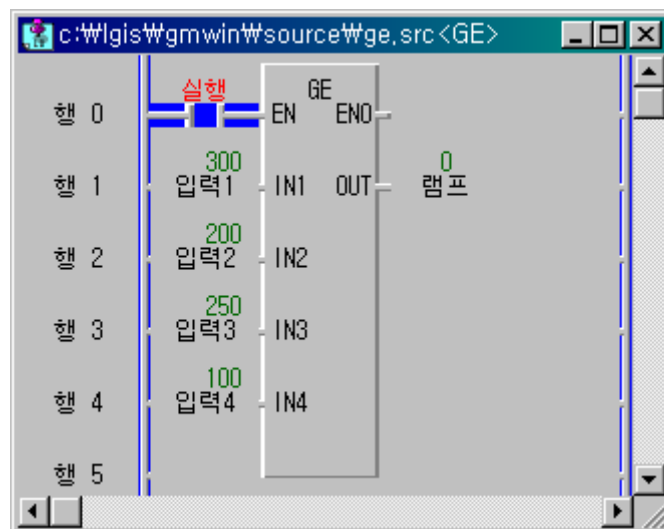
변수명	변수 종류	메모리할당	사용여부	데이터 타입
램프	VAR	%IX0.1.0		BOOL
실행	VAR	%IX0.0.0		BOOL
입력1	VAR	<자동>		INT
입력2	VAR	<자동>		INT
입력3	VAR	<자동>		INT
입력4	VAR	<자동>		INT

<프로그램 및 지역변수>



<모니터링 1>

입력 1>입력 2>입력 3>입력 4 의 조건을 만족하므로 OUT 단자에 1 이 출력되며, LED 가 ON 됩니다.



<모니터링 2>

입력 1>입력 2>입력 3>입력 4 의 조건을 만족하지 못하므로 OUT 단자에 0 이 출력되며, LED 가 OFF 됩니다.

EQ

평 선	설 명
	<p>입력:</p> <p>EN:EN 이 1 일 때 평선 실행</p> <p>IN1~IN8:비교할 데이터</p> <p>출력:</p> <p>ENO:평선이 수행되면 1 출력</p> <p>OUT:비교 결과가 저장될 영역</p> <p>IN1=IN2...= INn 을 만족하면 1 출력</p>

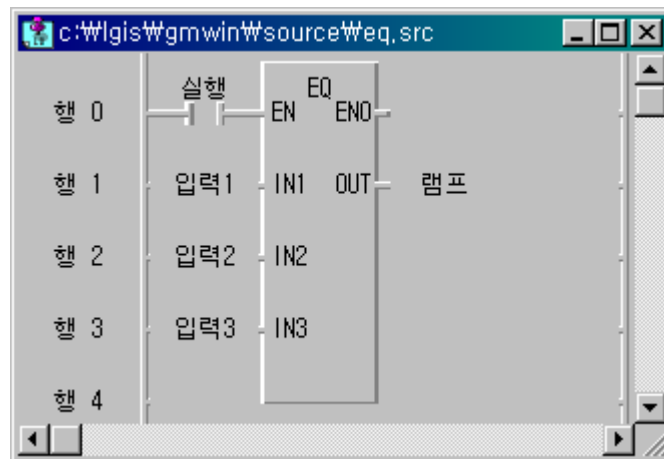
◎ 기능

입력값의 비교 결과 $IN1=IN2=IN3...=INn$ (n 은 입력 개수, 8 까지 가능)을 만족하면 OUT 으로 1 을 출력합니다.

입력값의 비교 결과 $IN1=IN2=IN3...=INn$ 을 만족하지 못하는 경우에는 OUT 으로 0 을 출력합니다.

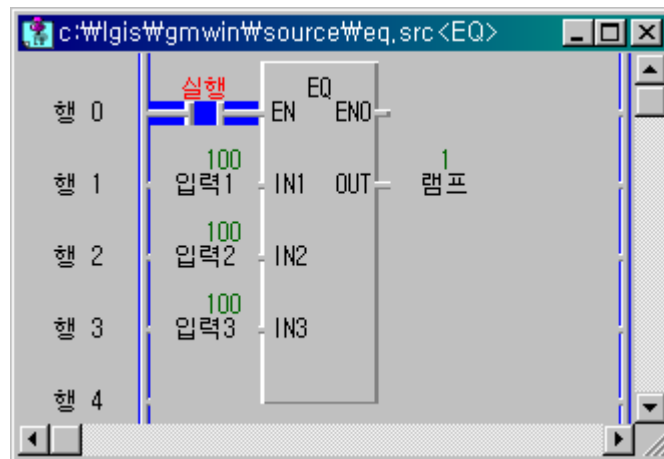
◎ 프로그램

세 개의 입력값이 동일할 때 LED 램프가 ON 됩니다.



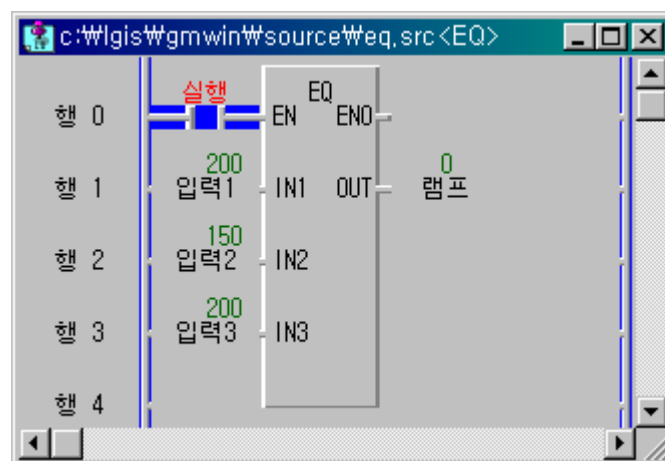
변수명	변수 종류	메모리할당	사용여부	데이터 타입
램프	VAR	%QX0.1.0		BOOL
실행	VAR	%IX0.0.0		BOOL
입력1	VAR	<자동>		INT
입력2	VAR	<자동>		INT
입력3	VAR	<자동>		INT

<프로그램 및 지역 변수>



<모니터링 1>

입력 1 = 입력 2 = 입력 3 의 조건을 만족하므로 OUT 단자에 1 이 출력되며, 램프가 ON 됩니다.



<모니터링 1>

입력 1 = 입력 2 = 입력 3 의 조건을 만족하지 못하므로 OUT 단자에 0 이 출력되며, 램프가 OFF 됩니다.

LE

평 선	설 명
	<p>입력:</p> <p>EN: EN 이 1 일 때 평선 실행</p> <p>IN1~IN8: 비교할 데이터</p> <p>출력:</p> <p>ENO: 평선이 수행되면 1 출력</p> <p>OUT: 비교 결과가 저장될 영역</p> <p>$IN1 \leq IN2 \leq \dots \leq INn$ 을 만족하면 1 출력</p>

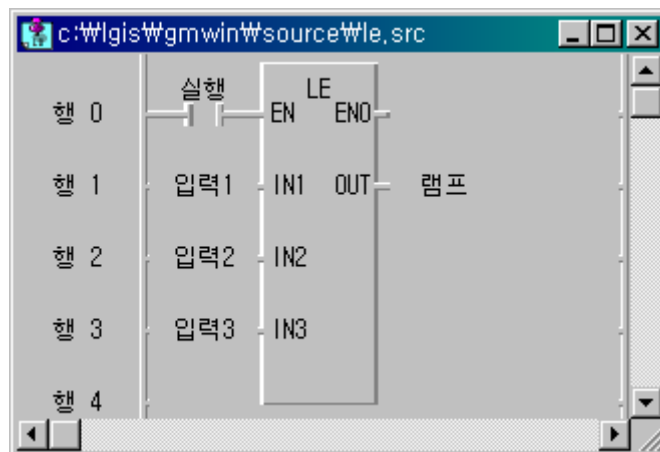
◎ 기능

입력값의 비교 결과 $IN1 \leq IN2 \leq IN3 \leq \dots \leq INn$ (n 은 입력 개수, 8 까지 가능)을 만족하면 OUT 으로 1 을 출력합니다.

입력값의 비교 결과가 $IN1 \leq IN2 \leq IN3 \leq \dots \leq INn$ 을 만족하지 못하면 OUT 으로 0 을 출력합니다.

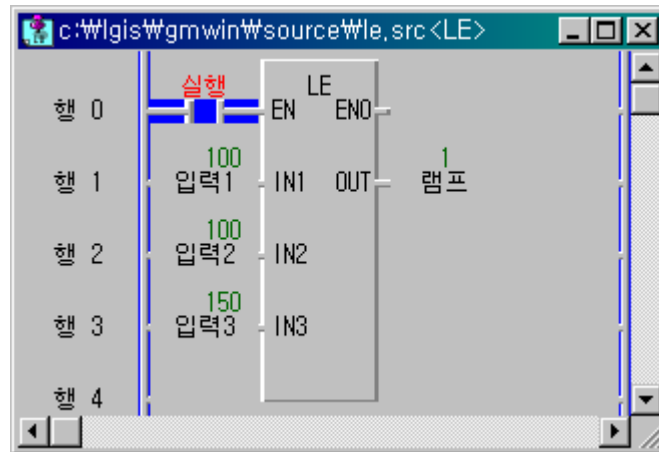
◎ 프로그램

세 개의 정수를 입력으로 받아 입력 1 ≤ 입력 2 ≤ 입력 3 을 만족하면 램프가 ON 됩니다.



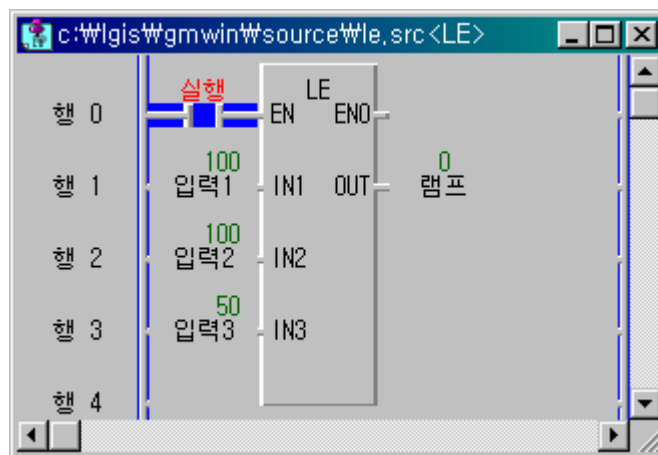
변수명	변수 종류	메모리 할당	사용여부	데이터 타입
램프	VAR	%QX0.1.0		BOOL
실행	VAR	%IX0.0.0		BOOL
입력1	VAR	<자동>		INT
입력2	VAR	<자동>		INT
입력3	VAR	<자동>		INT

<프로그램 및 지역 변수>



<모니터링 1>

입력 1≤입력 2≤입력 3의 조건을 만족하므로 OUT 단자에 1이 출력이 되며, 램프는 ON 됩니다.



<모니터링 2>

입력 1≤입력 2≤입력 3의 조건을 만족하지 못하므로 OUT 단자에 0이 출력이 되며, 램프는 OFF 됩니다.

LT

평 선	설 명
	<p>입력:</p> <p>EN: EN 이 1 일 때 평선 실행</p> <p>IN1~IN8: 비교할 데이터</p> <p>출력:</p> <p>ENO: 평선이 수행되면 1 출력</p> <p>OUT: 비교 결과가 저장될 영역</p> <p>IN1<IN2... <INn 을 만족하면 1 출력</p>

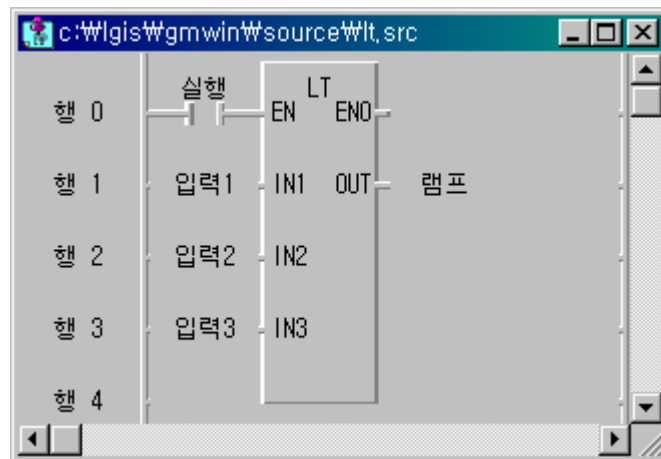
● 기능

입력값의 비교 결과 $IN1 < IN2 < IN3 \dots < INn$ (n 은 입력 개수, 8 까지 가능)을 만족하면 OUT 으로 1 을 출력합니다.

입력값의 비교 결과 $IN1 < IN2 < IN3 \dots < INn$ (n 은 입력 개수, 8 까지 가능)을 만족하지 않으면 OUT 으로 0 을 출력합니다.

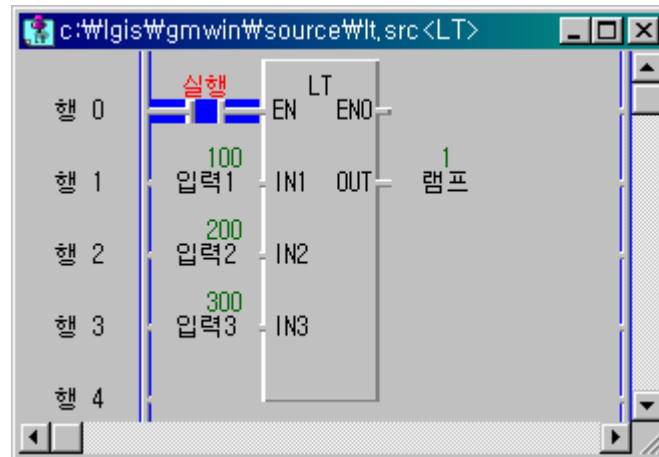
● 프로그램

세 개의 정수값을 비교하여 입력 1<입력 2<입력 3 의 조건을 만족하면 램프가 ON 됩니다.



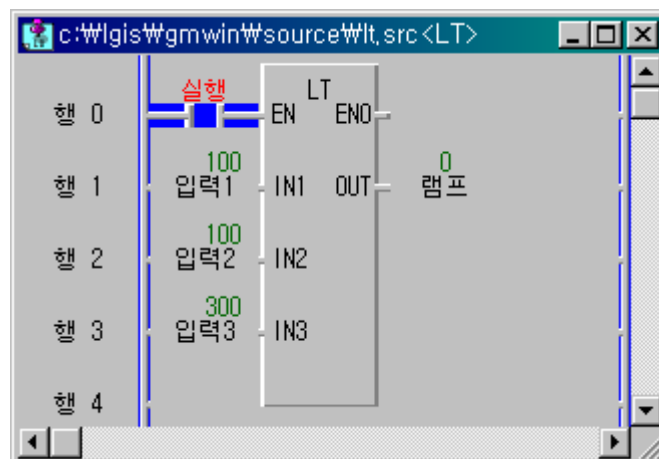
변수명	변수 종류	메모리할당	사용여부	데이터 타입
램프	VAR	%QX0.1.0		BOOL
실행	VAR	%IX0.0.0		BOOL
입력1	VAR	<자동>		INT
입력2	VAR	<자동>		INT
입력3	VAR	<자동>		INT

<프로그램 및 지역 변수>



<모니터링 1>

입력 1<입력 2<입력 3 을 만족하므로 OUT 단자에는 1 이 출력되고, 램프는 ON 됩니다.



<모니터링 2>

입력 1<입력 2<입력 3 을 만족하지 못하므로 OUT 단자에는 0 이 출력되고, 램프는 OFF 됩니다.

NE

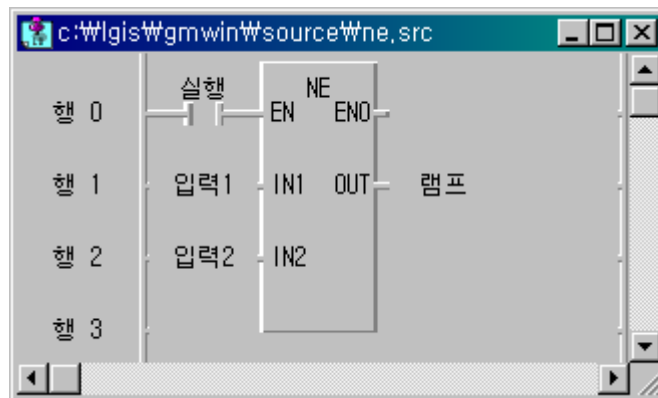
평 선	설 명
	<p>입력:</p> <p>EN:EN 이 1 일 때 평선 실행</p> <p>IN1, IN2 :비교할 데이터</p> <p>출력:</p> <p>ENO:평선이 수행되면 1 출력</p> <p>OUT:비교 결과 저장 영역</p> <p>IN1 ≠ IN2 이면 1 출력</p>

○ 기능

IN1 과 IN2 를 비교하여 그 결과가 같지 않으면 OUT 으로 1 을 출력합니다.
 IN1 과 IN2 를 비교하여 그 결과가 같으면 OUT 으로 0 을 출력합니다.

○ 프로그램

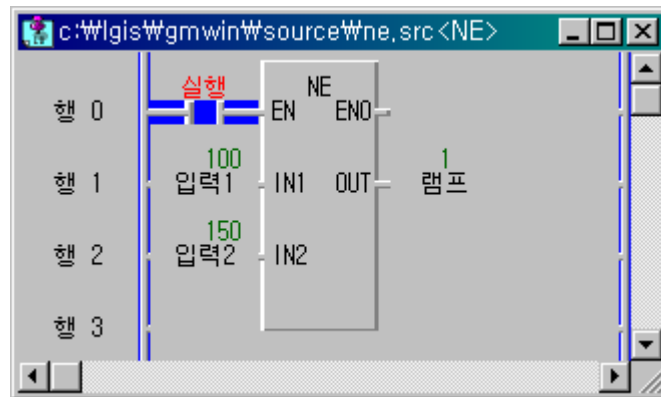
두 개의 정수 입력을 받아 비교하여 두 개의 값이 다르면 램프가 ON 되고
 두 개의 값이 같으면 램프가 OFF 됩니다.



<프로그램>

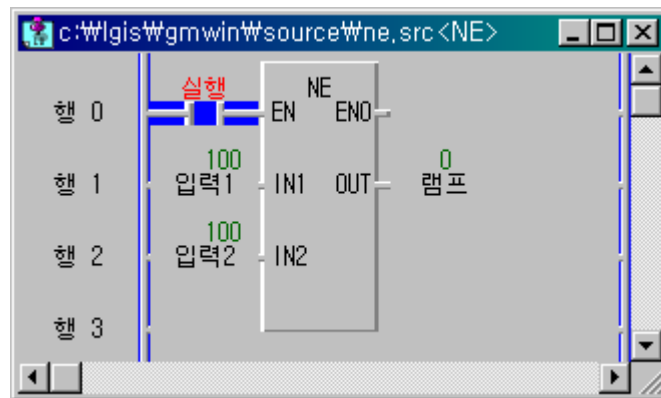
변수명	변수 종류	메모리할당	사용여부	데이터 타입
램프	VAR	%QX0.1.0		BOOL
실행	VAR	%IX0.0.0		BOOL
입력1	VAR	<자동>		INT
입력2	VAR	<자동>		INT

<지역변수>



<모니터링 1>

입력 1 과 입력 2 의 값이 다르므로 OUT 단자에는 1 이 출력되고, 램프는 ON 됩니다.



<모니터링 2>

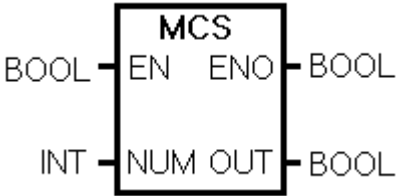
입력 1 과 입력 2 의 값이 같으므로 OUT 단자에는 0 이 출력되고, 램프는 OFF 됩니다.

◎ 비교 평선 프로그램 예



	변수 명	데이터 타입	메모리 할당	초기 값	변수 종류	사용 여부	설명문
1	값1	INT	<자동>	300	VAR	*	
2	값10	INT	<자동>	150	VAR		
3	값11	INT	<자동>	200	VAR		
4	값12	INT	<자동>	250	VAR		
5	값15	INT	<자동>	100	VAR		
6	값16	INT	<자동>	200	VAR		
7	값17	INT	<자동>	300	VAR		
8	값18	INT	<자동>	300	VAR		
9	값19	INT	<자동>	200	VAR		
10	값2	INT	<자동>	200	VAR	*	
11	값3	INT	<자동>	100	VAR	*	
12	값4	INT	<자동>	300	VAR	*	
13	값5	INT	<자동>	200	VAR	*	
14	값6	INT	<자동>	200	VAR	*	
15	값7	INT	<자동>	300	VAR	*	
16	값8	INT	<자동>	300	VAR	*	
17	값9	INT	<자동>	300	VAR	*	

5.2.3.5 마스터 컨트롤 (MCS/MCSCLR)

평 선	설 명
	<p>입력 EN : 1 일때 평선 실행 NUM : Nesting (0~15)</p> <p>출력 ENO : MCS 명령이 실행되면 1 을 출력 OUT : Dummy(항상 0 을 출력합니다.)</p>

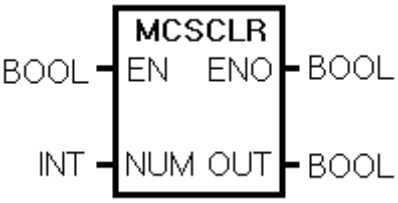
○ 기능

- ▷ EN 이 On 이면, Master Control 이 수행됩니다. 이경우, MCS 평선에서 MCSCLR 평선 사이의 프로그램은 정상적으로 수행됩니다.
- ▷ EN 이 Off 인경우, MCS 평선에서 MCSCLR 평선사이의 프로그램은 아래와 같이 수행됩니다.

명령어	명령어 상태
Timer	현재값은 0 이되고, 출력(Q)은 Off 됩니다.
Counter	출력(Q)은 Off 되고, 현재값은 현재 상태를 유지합니다.
코일	모두 Off 됩니다.
역코일	모두 Off 됩니다.
셋코일, 리셋코일	현재 값을 유지합니다.
평선, 평선 블록	현재 값을 유지합니다.

- ▷ EN 이 Off 인경우에도 MCS 평선에서 MCSCLR 평선 사이의 명령들이 위와 같이 수행되기 때문에 스캔 타임이 감소되지 않습니다.
- ▷ Master Control 명령은 Nesting 해서 사용될 수 있습니다. 즉, Master Control 영역이 Nesting(NUM)에 의해 구분될 수 있습니다. Nesting(NUM)은 0 에서 15 까지 설정이 가능하고, 만약 16 이상으로 설정한 경우 Master Control 이 정상적으로 동작하지 않습니다.

알아 두기 MCSCLR 없이 MCS 명령을 사용한 경우, MCS 평선에서 프로그램의 마지막 행까지 Master Control 이 수행되니 주의 바랍니다.
또, MCSCLR 평선 앞에는 접점을 사용하지 않습니다.

평	선	설	명
		입력 EN : 1 일때 평선 실행 NUM : Nesting (0~15)	출력 ENO : MCSCCLR 명령이 실행되면 1을 출력 OUT : MCSCCLR 명령이 실행되면 1을 출력

○ 기능

- ▷ Master Control 명령을 해제합니다. 그리고, Master Control 영역의 마지막을 가리킵니다.
- ▷ MCSCCLR 평선 동작 시 Nesting(NUM)의 값보다 같거나 작은 모든 MCS 명령을 해제합니다.

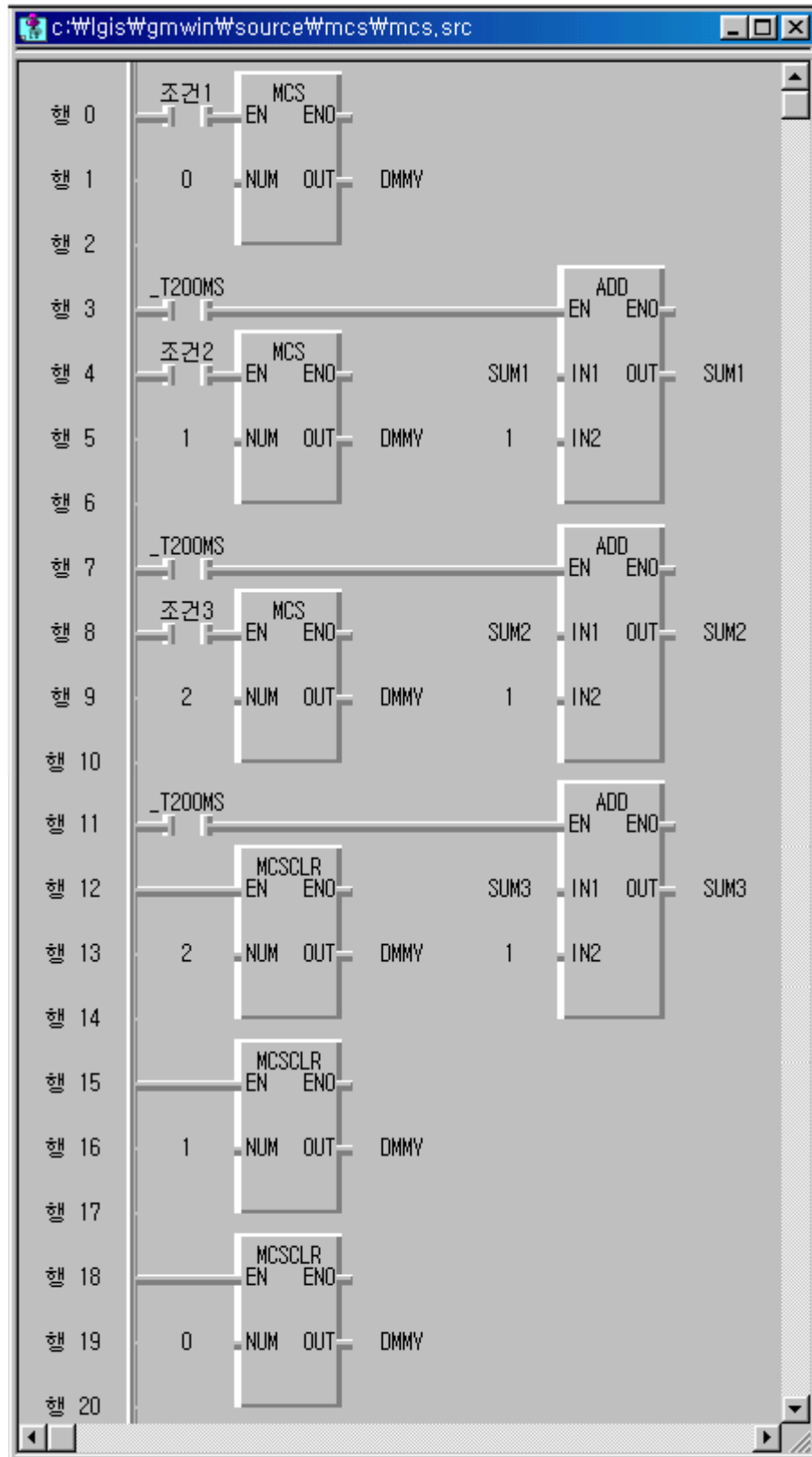
※ 네스팅(NESTING)이란 : 다중 마스터컨트롤 사용에 의한 제어를 의미합니다.



○ 프로그램 예

%IX0.0.0 스위치를 ON 시키면 200ms 마다 SUM1 이 1 씩 증가되고, %IX0.0.0 과 %IX0.0.1 을 동시에 ON 시키면 SUM1 과 SUM2 가 1 씩 증가되며, %IX0.0.0 과 %IX0.0.1, %IX0.0.2.를 동시에 ON 시키면 SUM1, SUM2, SUM3 이 200ms 마다 1 씩 증가합니다. %IX0.0.0 을 OFF 시키면 SUM1, SUM2, SUM3 은 데이터를 그대로 유지합니다.

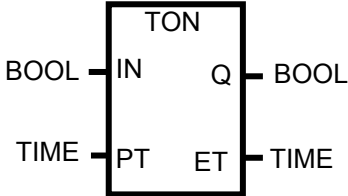
	변수명	데이터타입	메모리할당	초기값	변수종류	사용여부	설명문
1	조건1	BOOL	%IX0.0.0		VAR	*	
2	조건2	BOOL	%IX0.0.1		VAR	*	
3	조건3	BOOL	%IX0.0.2		VAR	*	
4	DMY	BOOL	<자동>		VAR	*	
5	SUM1	INT	<자동>		VAR	*	
6	SUM2	INT	<자동>		VAR	*	
7	SUM3	INT	<자동>		VAR	*	



5.3 평선 블록 프로그램

5.3.1 타이머

5.3.1.1 TON (ON-Delay Timer)

평 선	설 명
	입력: IN:타이머의 기동 조건 PT:설정 시간 출력: Q:타이머 접점 출력 CV:경과 시간

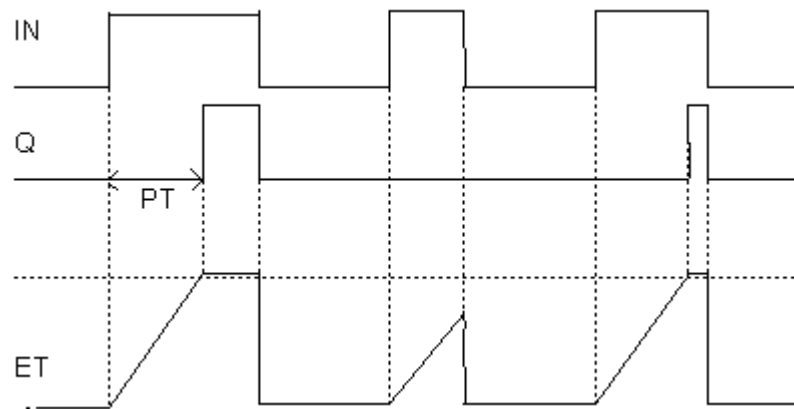
◎ 기능

IN 이 1 이 된 후 경과 시간이 ET 로 출력됩니다.

만일, 경과 시간 ET 가 설정 시간에 도달하기 전에 IN 이 0 이면, 경과 시간은 0 으로 됩니다.

Q 가 1 이 된 후 IN 이 0 이 되면, Q 는 0 이 됩니다.

◎ 타임 차트



◎ 프로그램

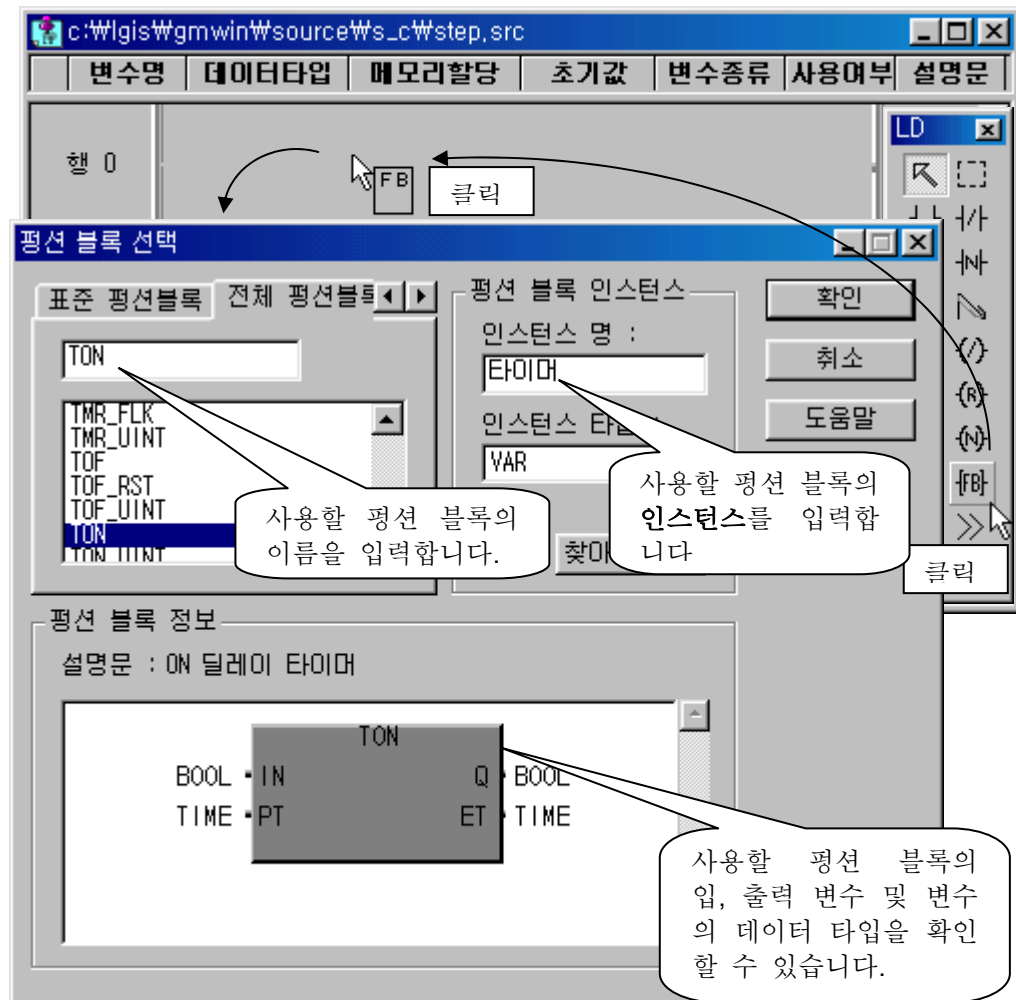
ON 스위치를 ON 시킨 후 5 초 이상 ON 상태를 유지시키면 5 초 후에 LED 램프가 ON 됩니다. 만일 ON 스위치를 5 초 이전에 OFF 시키면 경과 시간은 0 으로 리셋되고, LED 램프는 ON 되지 않습니다.

참고 사항 타이머의 최대 설정시간 : **T#49D17H2M47S295MS**

타이머의 설정 시간은 TIME 형으로 설정하며, 낱자는 D, 시간은 H, 분은 M, 초는 S, 1/1000 초는 MS 단위를 사용합니다.

예) 1 일 2 시간 3 분 4 초 567MS => T#1D2H3M4S567MS

- ◆ GMWIN 에서 프로젝트 및 프로그램 정의 후 프로그램 창에 평선 블록을 등록합니다. 평선 블록 등록 시 평선 블록 인스턴스를 등록해야 합니다.



참고 사항 평선 블록 인스턴스란 평선 블록에 관련된 입, 출력 변수를 통합적으로 관리하기 위해 설정하는 변수입니다.

c:\Wlgis\Wgmwin\source\Ws_c\step.src *

	변수명	데이터타입	메모리할당	초기값	변수종류	사용여부	설명문
1	타이머	FB Instance	<자동>		VAR		
2	LED_램프	BOOL	%QX0.1.0		VAR		
3	ON_스위치	BOOL	%IX0.0.0		VAR		

평선 블록 인스턴스 이름

평선 블록 이름

입력 변수는 평선 블록에서 지정된 고유의 데이터 타입을 설정해야 합니다.

평선 블록의 출력 변수는 반드시 설정하지 않아도 됩니다. 출력 변수를 설정하지 않을 경우, 인스턴스.변수로 자동할당 됩니다. 이 프로 그램에서 ET 는 타이머.ET 로 자동 등록 됩니다.

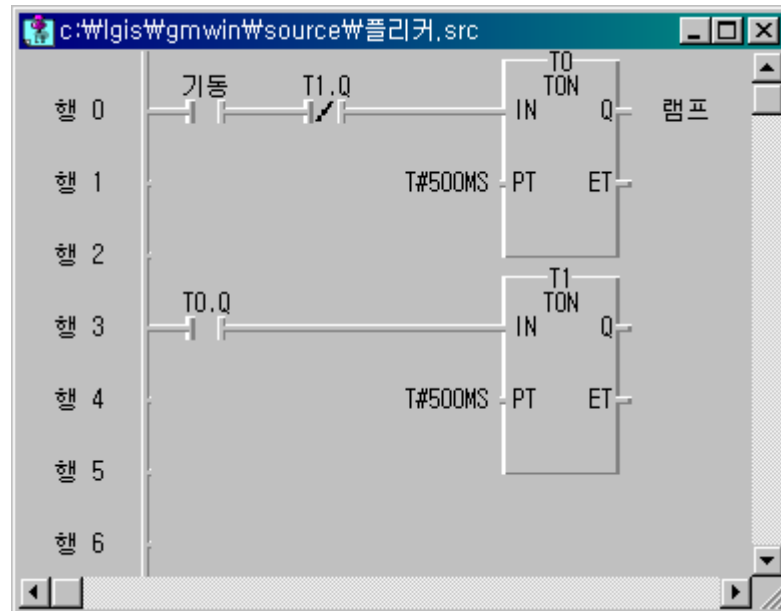
◎ 프로그램 예

- TON 은 평선 블록이므로 연산 중 누계되는 데이터를 잠시 보관하기 위한 인스턴스 변수를 반드시 선언해야 합니다.
- GMWIN 에서 프로그램 편집 시 TON 의 인스턴스 변수를 선언하면 타이머 출력은 인스턴스 이름.Q, 경과 시간은 인스턴스 이름.ET 로 변수가 자동 생성됩니다.
- ▷ TON 의 인스턴스 변수 T1 을 선언합니다.
- ▷ 타이머 T1 의 설정 시간을 7 초(T#7S)로 설정합니다.
- ▷ 기동 스위치 0 (%IX0.0.0)를 ON 하면 T1.ET 에 경과 시간이 표시됩니다.
- ▷ 경과 시간 T1.ET 가 설정 시간 7 초에 도달하면 타이머 출력 T1.Q 가 ON 됩니다.
- ▷ T1.Q 가 ON 된 후 기동 SW (%IX0.0.0)를 OFF 하면 T1.Q 는 OFF 됩니다.

	변수 명	데이터 타입	메모리 할당	초기 값	변수 종류	사용 여부	설명문
1	기동SW	BOOL	%IX0.0.0		VAR		
2	램프1	BOOL	%QX0.1.0		VAR		
3	T1	FB Instance	<자동>		VAR	*	

◎ **ON-Delay Timer** 를 이용한 프로그램 ; 플리커 회로

ON-Delay 타이머 두 개를 사용하여 램프를 플리커 시킵니다.



<프로그램>

변수명	변수 종류	메모리할당	사용여부	데이터 타입
기동	VAR	%IX0.0.0	*	BOOL
램프	VAR	%QX0.1.0	*	BOOL
T0	VAR	<자동>	*	FB Instance
T1	VAR	<자동>	*	FB Instance

<지역변수>

5.3.1.2 TOF (OFF-Delay Timer)

평 섯	설 명
	<p>입력: IN:타이머의 기동 조건 PT:설정 시간</p> <p>출력: Q:타이머 접점 출력 CV:경과 시간</p>

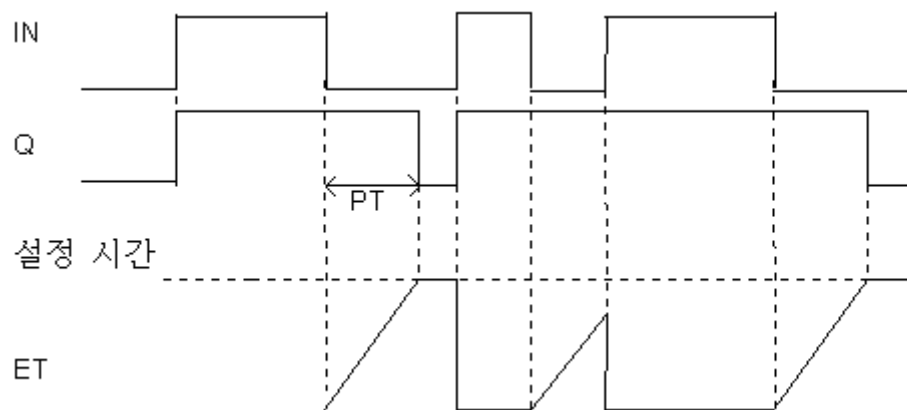
◎ 기능

IN 이 ON 되면, Q 가 ON 되고, IN 이 OFF 된 후부터 PT 에 의해서 지정된 설정 시간이 경과한 후 Q 가 OFF 됩니다.

IN 이 OFF 된 후 경과 시간이 ET 로 출력됩니다.

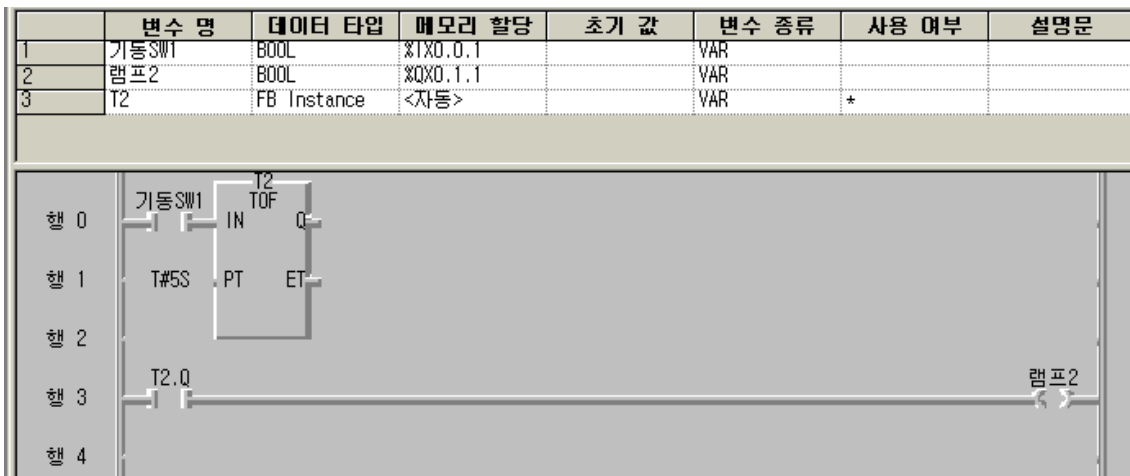
만일 경과 시간 ET 가 설정 시간에 도달하기 전에 IN 이 ON 되면, 경과 시간은 다시 0 으로 됩니다.

◎ 타임 차트



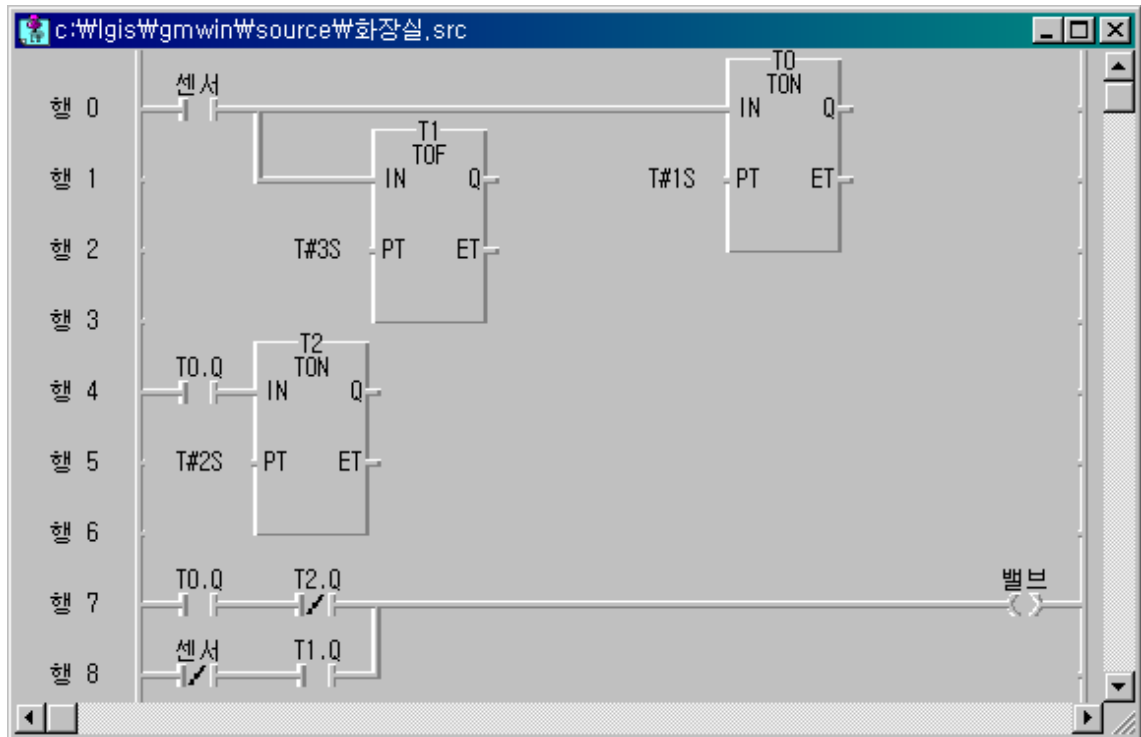
◎ 프로그램 예

- ☞ TOF 는 펄스 블록이므로 연산 중 누계되는 데이터를 잠시 보관하기 위한 인스턴스 변수를 반드시 선언해야 합니다.
- ☞ GMWIN 에서 프로그램 편집 시 TOF 의 인스턴스 변수를 선언하면 타이머 출력은 인스턴스 이름.Q, 경과 시간은 인스턴스 이름.ET 로 변수가 자동 생성됩니다.
- ▷ TOF 의 인스턴스 변수 T2 를 선언합니다.
- ▷ 타이머 T2 의 설정 시간을 5 초(T#5S)로 설정합니다.
- ▷ 기동 SW1 (%IX0.0.1)을 ON 하면 타이머 출력 T2.Q 가 ON 됩니다.
- ▷ 기동 SW1 (%IX0.0.1)을 OFF 하면 T2.ET 에 경과 시간이 표시됩니다.
- ▷ 경과 시간 T2.ET 가 설정시간 5 초에 도달하면 타이머 출력 T2.Q 가 OFF 됩니다.



◎ TON, TOF 를 이용한 프로그램 ; 화장실 자동 밸브 제어

사용자가 변기에 접근한 후 1 초 뒤 2 초간 물이 나오고 이탈 후 즉시 3 초간 물이 공급되는 회로입니다.

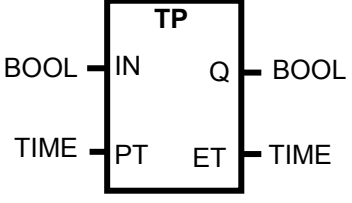


<프로그램>

변수명	변수 종류	메모리할당	사용여부	데이터 타입
밸브	VAR	%QX0.1.0	*	BOOL
센서	VAR	%IX0.0.0	*	BOOL
T0	VAR	<자동>	*	FB Instance
T1	VAR	<자동>	*	FB Instance
T2	VAR	<자동>	*	FB Instance

<지역변수>

5.3.1.3 TP (Pulse Timer)

평 선	설 명
	<p>입력: IN:타이머의 기동 조건 PT:설정 시간</p> <p>출력: Q:타이머 접점 출력 CV:경과 시간</p>

● 기능

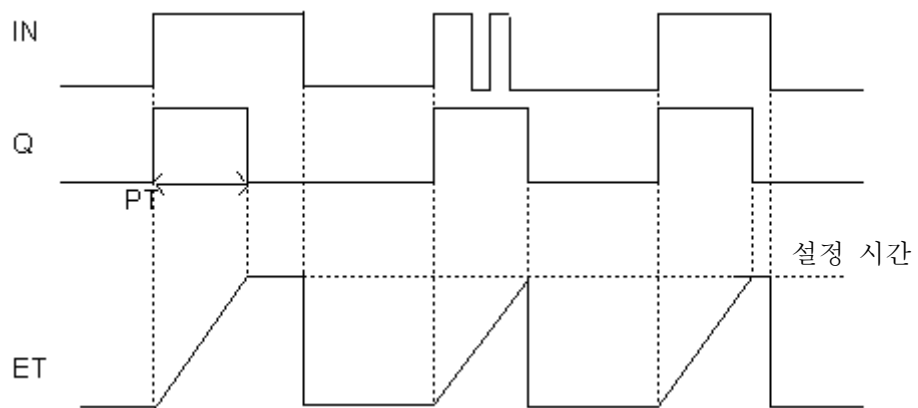
IN 이 ON 되면 PT 에 의해서 지정된 설정 시간 동안만 Q 가 ON 되고, ET 가 PT 에 도달하면 자동으로 0 이 됩니다.

경과 시간 ET 는 IN 이 ON 되었을 때부터 증가하며 PT 에 이르면 값을 유지하다가 IN 이 0 이 될 때 0 의 값이 됩니다.

ET 가 증가할 동안 IN 이 OFF 되거나 다시 ON 되어도 영향이 없습니다.

ET 가 PT 에 도달한 후 IN 이 다시 ON 되면 ET 가 증가합니다.

● 타임 차트



◎ 프로그램 예

☞ TP 는 펄스 블록이므로 연산 중 누계되는 데이터를 잠시 보관하기 위한 인스턴스 변수를 반드시 선언해야 합니다.

☞ GMWIN 에서 프로그램 편집 시 TP 의 인스턴스 변수를 선언하면 타이머 출력은 인스턴스 이름.Q, 경과 시간은 인스턴스 이름.ET 로 변수가 자동 생성됩니다.

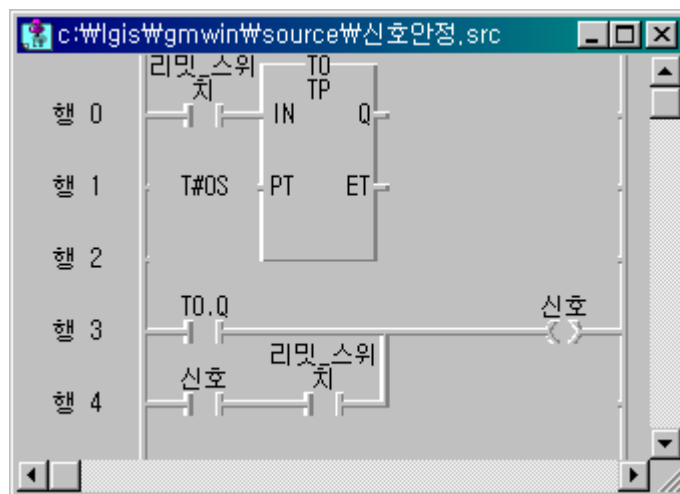
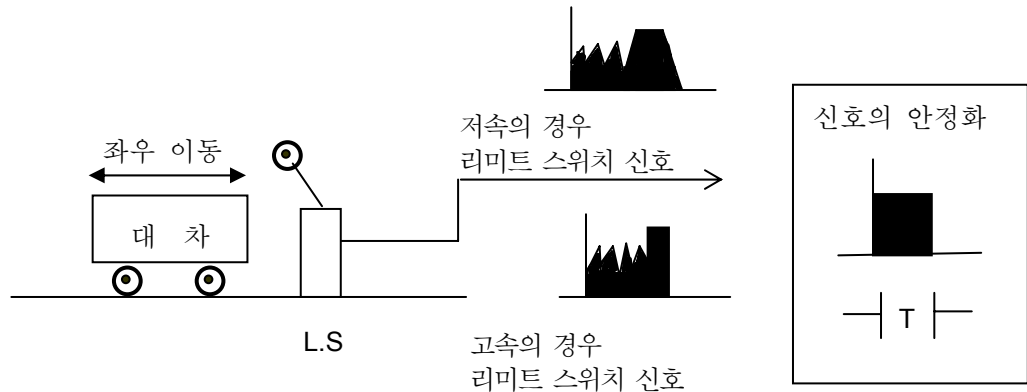
- ▷ TP 의 인스턴스 변수 T3 를 선언합니다.
- ▷ 타이머 T3 의 설정 시간을 5 초(T#5S)로 설정합니다.
- ▷ 기동 SW0 (%IX0.0.0)를 OFF → ON 하면 타이머 출력 T3.Q 가 5 초 동안 ON 했다 OFF 합니다.
- ▷ T3.ET 가 증가할 동안 기동 스위치 0 가 OFF 되거나 다시 ON 되어도 영향이 없습니다.
- ▷ T3.ET 가 증가할 동안 T3.ET 에 경과 시간이 표시됩니다.



◎ TP 를 이용한 프로그램 ; 신호떨림 방지 회로

속도가 일정치 않은 물체의 통과신호(리미트 스위치)의 떨림을 방지하여, 안정된 신호를 얻습니다.

◎ 시스템 도



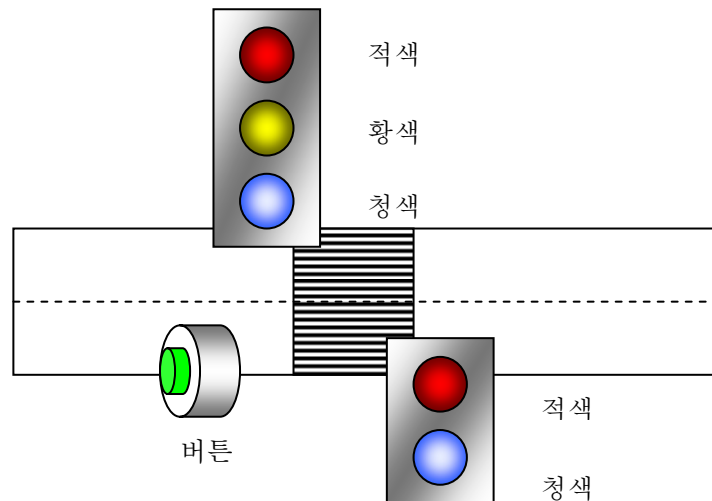
<프로그램>

변수명	변수 종류	메모리할당	사용여부	데이터 타입
리미트 스위치	VAR	%IX0.0.0	*	BOOL
신호	VAR	%QX0.1.0	*	BOOL
TO	VAR	<자동>	*	FB Instance

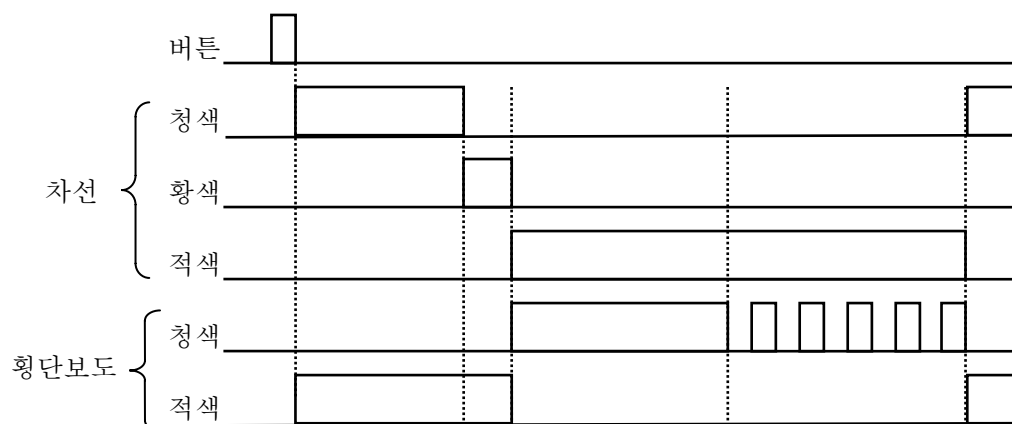
<지역변수>

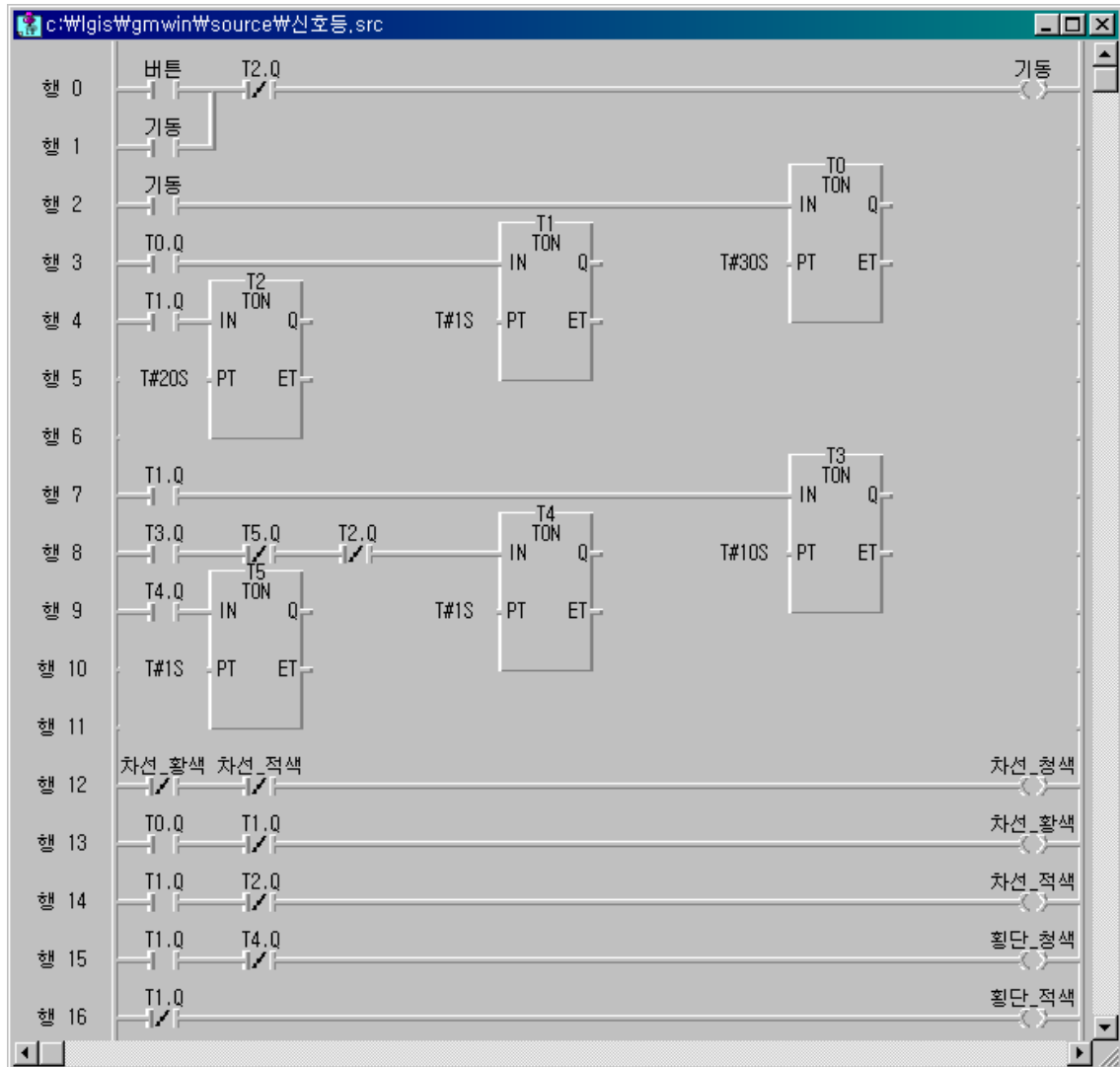
◎ TON 을 이용한 프로그램 ; 신호등 제어

보행자가 보행버튼을 누르면 30 초 후 차선의 신호등은 황색램프가 점등되며 1 초 후 적색으로 바뀝니다. 이때 보행자신호등은 청색램프가 10 초간 점등된 뒤 10초간 점멸하며 이후 적색으로 바뀝니다.



◎ 타임차트





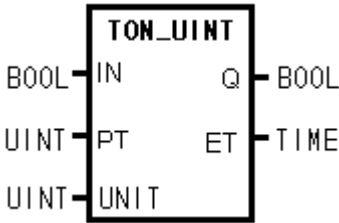
<프로그램>

변수명	변수 종류	메모리할당	사용여부	데이터 타입
기동	VAR	<자동>	*	BOOL
버튼	VAR	%IX0.0.0	*	BOOL
차선_적색	VAR	%QX0.1.2	*	BOOL
차선_청색	VAR	%QX0.1.0	*	BOOL
차선_활색	VAR	%QX0.1.1	*	BOOL
횡단_적색	VAR	%QX0.1.4	*	BOOL
횡단_청색	VAR	%QX0.1.3	*	BOOL

<지역변수>

5.3.2 응용 타이머

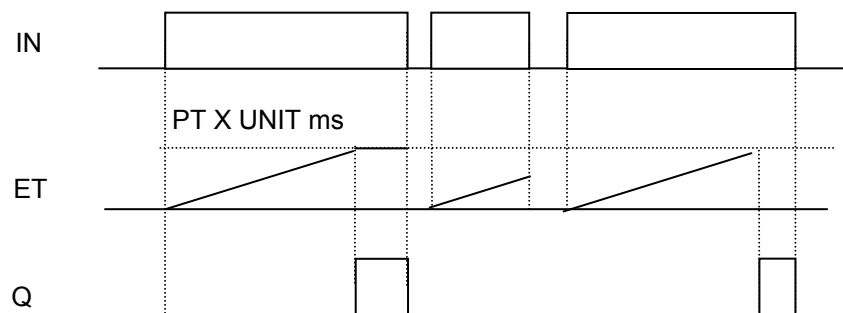
5.3.2.1 TON_UINT *

평 선	설 명
	<p>입력:</p> <p>IN:타이머의 기동 조건</p> <p>PT:설정 시간</p> <p>UNIT:설정 시간의 단위</p> <p>출력:</p> <p>Q:타이머 점점 출력</p> <p>ET:경과 시간</p>

◎ 기능

- ▷ TON_UINT 평선 블록은 IN 이 ON 된 후 경과시간이 ET 로 출력됩니다.
- ▷ 만일 경과시간 ET 가 설정시간에 도달하기 전에 IN 이 OFF 되면, 경과 시간 ET 는 0 으로 됩니다.
- ▷ Q 가 ON 된 후 IN 이 OFF 되면, Q 는 OFF 됩니다.
- ▷ 설정시간은 $PT \times UNIT[ms]$ 입니다.

◎ 타임차트



알아두기 정수 설정 타이머를 사용하고자 할 경우 GMWIN 의 라이브러리에서 APP.xFB 를 등록해야 사용할 수 있습니다.
본 교재에서 *표시된 평선 블록은 모드 APP.xFB 를 등록해야 합니다.

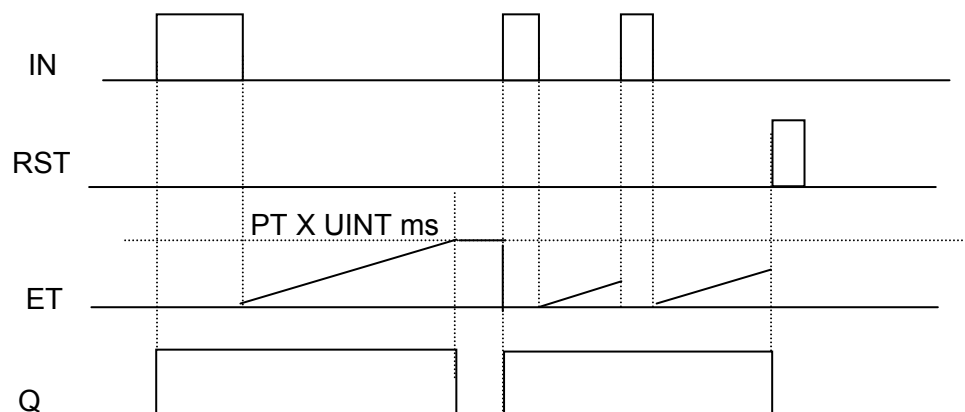
5.3.2.2 TOF_UINT *

평 선택	설 명
	<p>입력:</p> <p>IN:타이머의 기동 조건</p> <p>PT:설정 시간</p> <p>UNIT:설정 시간의 단위</p> <p>RST:리셋 입력</p> <p>출력:</p> <p>Q:타이머 접점 출력</p> <p>ET:경과 시간</p>

◎ 기능

- ▷ TOF_UINT 평선 블록은 기동 조건 IN 이 ON 되는 순간 Q 는 ON 되고, IN 이 OFF 된 후부터 PT 에 의하여 지정된 설정시간이 경과한 후 Q 가 OFF 됩니다.
- ▷ IN 이 OFF 된 후 경과 시간이 ET 로 출력됩니다.
- ▷ 만일 경과시간 ET 가 설정시간에 도달하기 전에 IN 이 ON 되면, 경과 시간은 다시 0 으로 됩니다.
- ▷ Reset 입력 조건이 성립하면 타이머 출력 Q 는 OFF 되고 경과 시간도 0 이 됩니다.
- ▷ 설정시간은 $PT \times UNIT[ms]$ 입니다.

◎ 타임차트



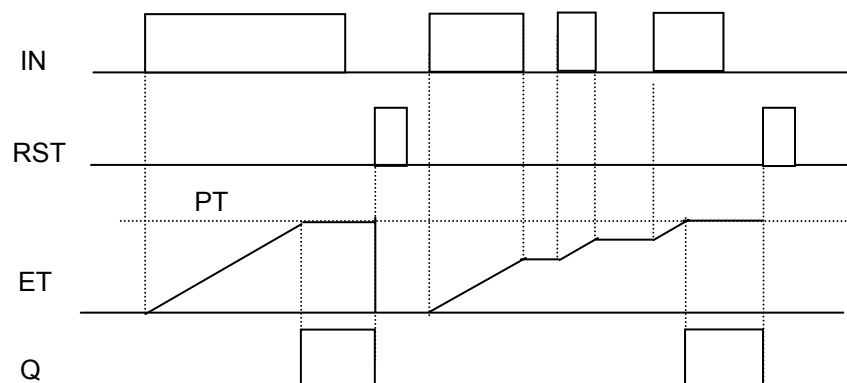
5.3.2.3 TMR *

평 섂	설 명
<p>The diagram shows a rectangular block labeled 'TMR'. On the left side, there are three inputs: 'IN' (labeled 'BOOL'), 'PT' (labeled 'TIME'), and 'RST' (labeled 'BOOL'). On the right side, there are two outputs: 'Q' (labeled 'BOOL') and 'ET' (labeled 'TIME').</p>	<p>입력:</p> <p>IN:타이머의 기동 조건 PT:설정 시간 RST:리셋 입력</p> <p>출력:</p> <p>Q:타이머 접점 출력 CV:경과 시간</p>

◎ 기능

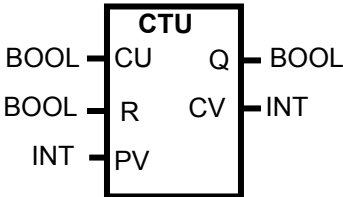
- ▷ TMR 평선 블록은 IN 이 ON 된 후 경과 시간이 ET 로 출력됩니다.
- ▷ 경과 시간 ET 가 설정시간에 도달하기 전에 IN 이 OFF 되어도 현재의 경과 시간을 유지하다가 IN 이 다시 ON 되면 경과 시간을 다시 증가시킵니다.
- ▷ 경과 시간이 설정 시간에 도달하면 Q 가 ON 됩니다.
- ▷ Reset 입력 조건이 성립되면 Q 는 OFF 되고 경과시간도 0 이 됩니다.

◎ 타임 차트



5.3.3 카운터

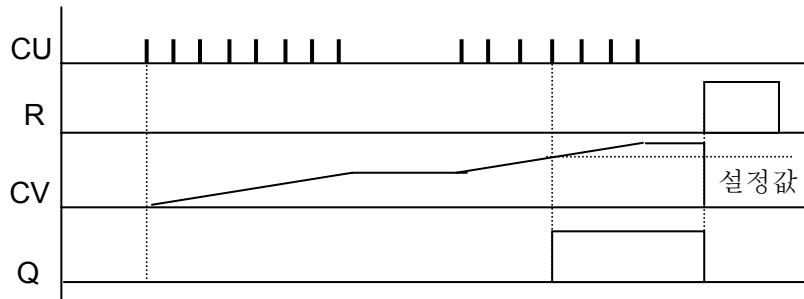
5.3.3.1 CTU(업 카운터)

평 선	설 명
	<p>입력:</p> <p>CU:펄스 신호 입력 R: 리셋 신호 입력 PV:설정값</p> <p>출력:</p> <p>Q:카운터 점점 신호 출력 CV:카운터 현재값 출력</p>

◎ 기능

- ▷ 펄스입력 CU 가 OFF → ON 으로 변하면 현재값 CV 가 이전값 보다 1 만큼 증가합니다. CV 는 정수(INT)의 최대값 32767 을 넘지 않습니다.
- ▷ 셋 입력 R 이 ON 되면 현재값 CV 는 0 으로 소거(Clear)됩니다.
- ▷ 현재값이 설정값과 같아지게 되면 카운터 출력 점점(Q)가 ON 되며 현재값이 설정값보다 큰 동안 ON 상태를 유지합니다.

◎ 타임 차트



◎ 프로그램

카운터의 설정값을 정수(INT)로 설정 합니다.
스위치가 ON 될 때 마다 카운터의 현재값이 증가하게 되며 카운터 현재값이 설정값 이상이면 램프가 ON 됩니다.

참고 사항

평선 블록은 여러 스캔에 걸쳐 출력을 만들기 때문에 평선 블록을 사용할 때는 연산 중 누계되는 데이터를 잠시 보관하기 위한 **인스턴스 변수**를 반드시 선언해야 합니다.

GMWIN 에서 프로그램 편집 시 카운터 인스턴스 변수를 선언하면 카운터 출력은 인스턴스 이름.Q, 현재값은 인스턴스 이름.CV 로 변수가 자동 생성됩니다.

◎ 프로그램 예

- ☞ CTU 는 펄선 블록이므로 연산 중 누계되는 데이터를 잠시 보관하기 위한 인스턴스 변수를 반드시 선언해야 합니다.
- ☞ GMWIN 에서 프로그램 편집 시 CTU 의 인스턴스 변수를 선언하면 카운터 출력은 인스턴스 이름.Q, 현재값은 인스턴스 이름.CV 로 변수가 자동 생성됩니다.
- ▷ CTU 의 인스턴스 변수 C1 을 선언합니다.
- ▷ 토글 스위치 0 (%IX0.0.0)로 CU 에 상승(Rising Edge) 펄스를 입력하면 현재값이 증가 합니다.
- ▷ 현재값을 우측 디지털 표시기(%QW0.1.1)에 출력합니다.
- ▷ 현재값이 설정값 이상이면 카운터 출력(C1.Q)이 1 이 되어 램프(%QX0.1.0)가 점등됩니다.
- ▷ 토글 스위치 1 (%IX0.0.1)을 ON 하면 현재값 및 카운터 출력이 리셋되어 0 이 됩니다.
- ▷ 현재값(C1.CV)이 0 ~ 9999 사이를 벗어나면 펄선 INT_TO_BCD 에 의해 _ERR, _LER 플래그가 ON 됩니다.

	변수 명	데이터 타입	메모리 할당	초기 값	변수 종류	사용 여부	설명문
1	램프	BOOL	%QX0.1.0		VAR		
2	표시기_右	WORD	%QW0.1.1		VAR		
3	C1	FB Instance	<자동>		VAR	*	

설명문

* 카운터 C1의 설정값 10을 입력합니다.

설명문

* 토글 스위치1(%IX0.0.0)로 펄스 입력을 하여 현재값(C1.CV)을 증가 시킵니다.

설명문

* 현재값(C1.CV)을 우측 디지털 표시기(%QW0.1.1)에 출력합니다.

설명문

* 현재값이 설정값 이상이면 카운터 출력(C1.Q)이 ON 되어 램프(%QX0.1.0)를 점등합니다.

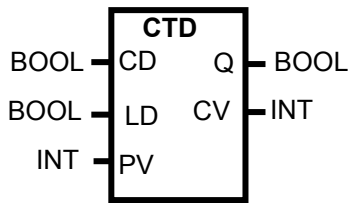
설명문

* 토글 스위치2(%IX0.0.1)를 ON 하면 리셋이 되어 현재값(C1.CV) 및 카운터 출력(C1.Q)은 0 이 됩니다.

```

        행 6  %IX0.0.0  CTU
                CU
        행 7  %IX0.0.1  R
                CV
        행 8    10    PV
        행 9
        행 10 INT_TO_BCD
                EN  END  C1.Q  램프
        행 11 C1.CV  IN1  OUT  표시기_右
        행 12
    
```

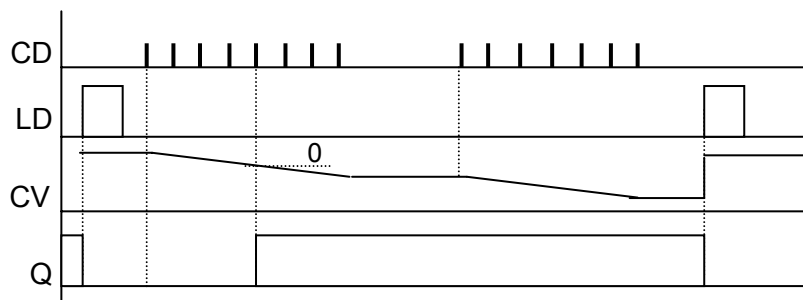
5.3.3.2 CTD (다운 카운터)

평 선	설 명
	<p>입력:</p> <p>CD:펄스 신호 입력 LD:설정값 입력 PV:설정값</p> <p>출력:</p> <p>Q:카운터 접점 신호 출력 CV:카운터 현재값 출력</p>

● 기능

- ▷ LD 단자를 ON 시키면 설정값이 현재값으로 로드 됩니다.
- ▷ 펄스입력 CD 가 OFF → ON 으로 변하면 현재값 CV 가 이전값 보다 1 만큼 감소합니다. 단, CV 는 정수(INT)의 최소값 -32768 을 넘지 않습니다.
- ▷ 현재값이 0 이되면 카운터 출력 접점(Q)가 ON 되며 현재값이 0 보다 작으면 ON 상태를 유지합니다.

● 타임 차트



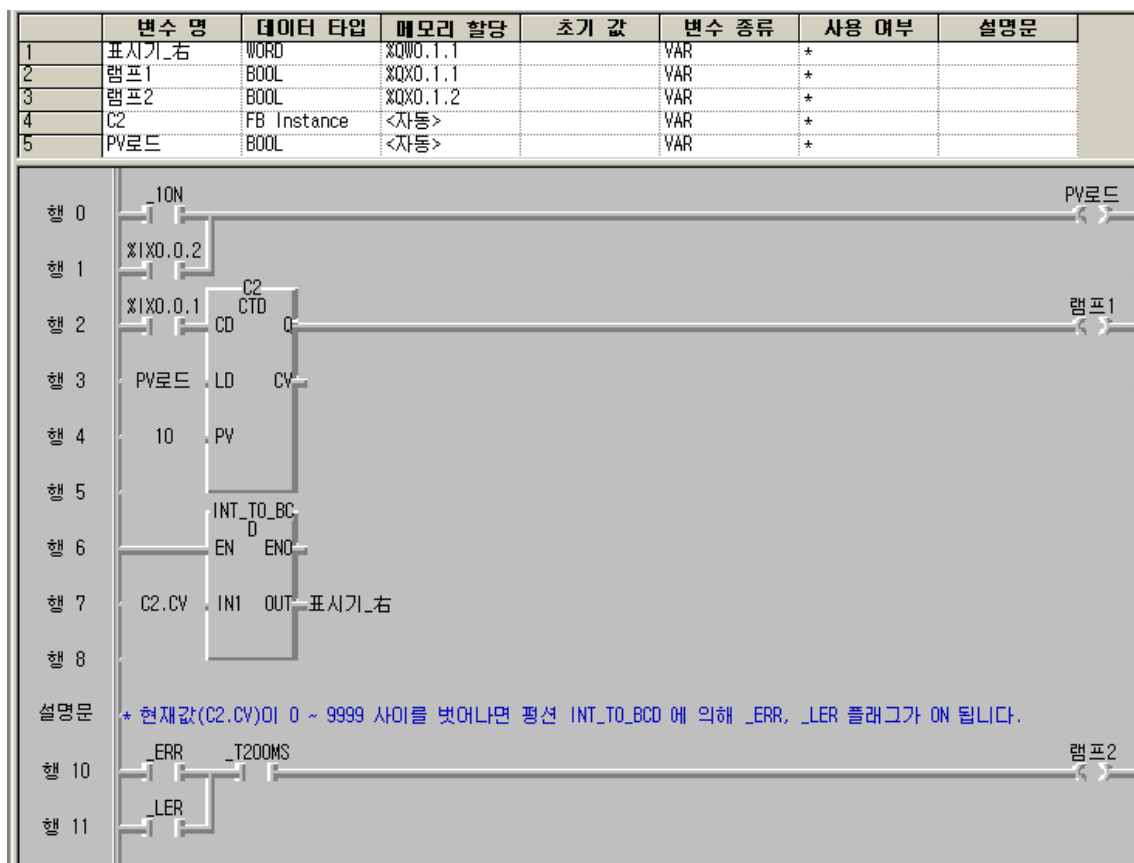
● 프로그램

정수(INT)로 카운터의 설정값을 설정하고 로드 스위치를 ON 하면 카운터의 설정값이 현재값으로 로드 됩니다.

토글 스위치가 ON 될 때 마다 카운터의 현재값이 감소하게 되며 카운터 현재값이 0 보다 작거나 같으면 램프가 ON 됩니다.

◎ 프로그램 예

- ▷ CTD의 인스턴스 변수 C2를 선언하고, 설정값을 10으로 셋팅합니다.
- ▷ 초기에 _1ON(첫 스캔 ON)에 의해 LD가 1이 되어 설정값이 현재값에 로드됩니다.
- ▷ 토글 스위치 1(%IX0.0.1)로 CD에 상승(Rising Edge) 펄스를 입력하면 현재값이 감소합니다.
- ▷ 현재값이 0 이하이면 카운터 출력(C2.Q)이 1이 되어 램프 1(%QX0.1.1)이 점등됩니다.
- ▷ 토글 스위치 2(%IX0.0.2)를 ON 하면 LD가 1이 되어 설정값이 현재값에 로드됩니다.



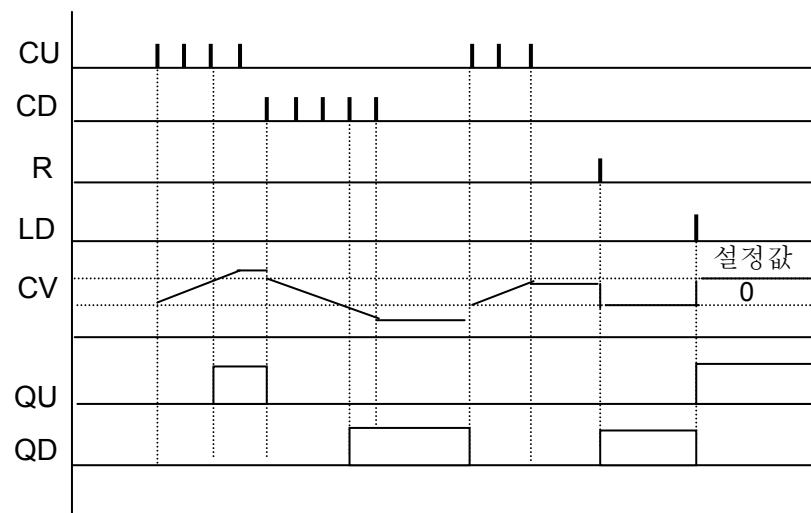
5.3.3.3 CTUD (업-다운 카운터)

평 선택	설 명
	<p>입력:</p> <p>CU:업 카운터 펄스 입력 CD:다운 카운터 펄스 입력 R :리셋 신호 입력 LD:설정값 입력 PV:설정값</p> <p>출력:</p> <p>QU:업 카운터 점점 출력 QD:다운 카운터 점점 출력 CV:카운터 현재값 출력</p>

◎ 기능

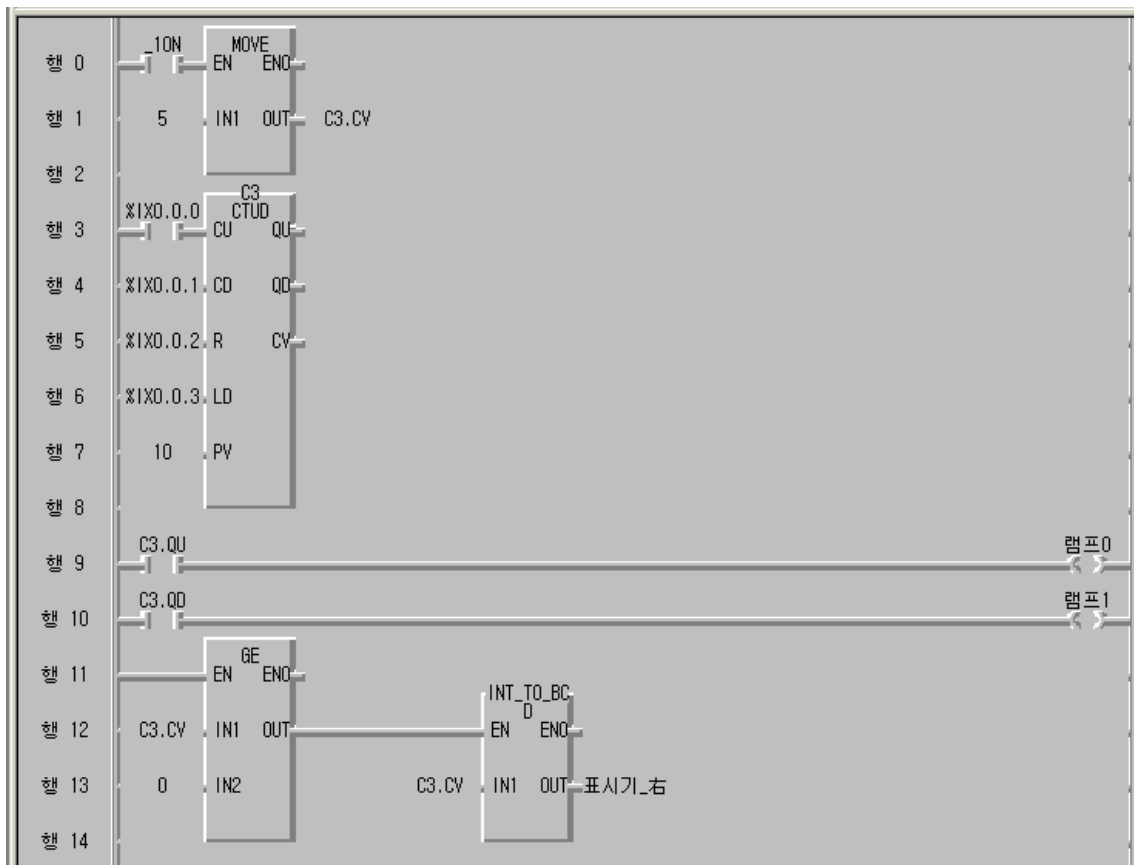
- ▷ CU 가 OFF→ON 되면 현재값 CV 가 이전값 보다 1 만큼 증가하고, CD 가 OFF→ON 되면 현재값 CV 가 이전값 보다 1 만큼 감소합니다. 단, 현재값 CV 는 정수(INT)의 최소값 -32768 ~ 최대값 32767 사이의 값을 갖습니다.
- ▷ 설정값 입력 점점 LD 가 ON 되면 현재값 CV 에 설정값 PV 값이 로드 됩니다. (CV=PV)
- ▷ 설정값 입력 R 이 ON 되면 현재값 CV 는 0 으로 클리어 됩니다.
- ▷ 출력 QU 는 CV 가 PV 이상 이면 ON 되고, QD 는 CV 가 0 이하 일 때 ON 됩니다.
- ▷ 각 입력 신호에 대해서 R > LD > CU > CD 순으로 동작을 수행하며, 신호의 중복 발생시 우선 순위가 높은 동작 하나만 수행합니다.

◎ 타임 차트



◎ 프로그램 예

- ▷ 초기에 _10N(첫 스캔 ON)에 의해 5가 C3.CV에 전송되어 현재값은 5가 됩니다.
- ▷ 토글 스위치 0(%IX0.0.0)로 CU에 상승 펄스를 입력하면 현재값이 증가 합니다.
- ▷ 토글 스위치 1(%IX0.0.1)로 CD에 상승 펄스를 입력하면 현재값이 감소 합니다.
- ▷ 현재값이 설정값 이상이면 C3.QU가 1이 되어 램프 0(%QX0.1.0)이 점등 됩니다.
- ▷ 현재값이 0 이하이면 C3.QD가 1이 되어 램프 1(%QX0.1.1)이 점등 됩니다.
- ▷ 토글 스위치 2(%IX0.0.2)를 ON 하면 리셋되어 현재값은 0으로 클리어(Clear) 됩니다.
- ▷ 토글 스위치 3(%IX0.0.3)를 ON 하면 LD가 1이 되어 설정값이 현재값에 로드 됩니다.
- ▷ 현재값이 0 이상 이면 GE 평선 출력이 ON 되어 평선 INT_TO_BCD가 실행됩니다. 단, 현재값 0 ~ 9999 사이를 벗어나면 평선 INT_TO_BCD에 의해 _ERR, _LER 플래그가 ON 됩니다.



	변수 명	데이터 타입	메모리 할당	초기 값	변수 종류	사용 여부	설명문
1	램프0	BOOL	%QX0.1.0		VAR	*	
2	램프1	BOOL	%QX0.1.1		VAR	*	
3	표시기_右	WORD	%QW0.1.1		VAR	*	
4	C3	FB Instance	<자동>		VAR	*	

5.3.4 순차 제어 (Step Controller)

5.3.4.1 SCON *

평 선택	설 명
	<p>입력</p> <p>REQ : 1 일때 평선 블록 실행</p> <p>S/O : 0 이면 SET 동작을 지정 1 이면 OUT 동작을 지정</p> <p>SET : 스텝의 번호(0~99)</p> <p>출력</p> <p>DONE : 평선 블록 실행이 완료 되면 ON.</p> <p>S : Set 된 bit array</p> <p>CUR_S : 현재 스텝 번호 출력</p>

● 기능

▷ 순차작업 조의 설정

- ☞ 평선 블록의 인스턴스 이름이 하나의 순차작업 조의 이름이 됩니다.
평선 블록 선언 예: S00, G01, 제조 1
스텝 점점 예: S00.S[1], G01.S[1], 제조 1.S[1]

▷ SET 동작일 경우(ST_0/JP_1 = 0)

- ☞ 동일 조 내에서 바로 이전의 스텝 번호가 On 되었을때 현재 스텝 번호가 On 됩니다.
- ☞ 현재 스텝 번호가 On 되면 자기 유지되어 입력 점점이 Off 되어도 On 되어진 상태를 유지합니다.
- ☞ 입력 조건 점점이 동시에 On 되어도 한 조 내에서는 한 스텝 번호만이 On 됩니다.
- ☞ Sxx.S[0]가 On 되면 모든 SET 출력이 Clear 됩니다.

▷ JUMP 동작일 경우(ST_0/JP_1 = 1)

- ☞ 동일 조 내에서 입력조건 점점이 다수가 On 하여도 한 개의 스텝 번호만 On 합니다.
- ☞ 입력 조건이 동시에 On 하면 나중에 프로그램 된 것이 우선으로 출력 됩니다.
- ☞ 현재 스텝번호가 On 되면 자기 유지되어 입력 조건이 Off 되어도 On 되어진 상태를 유지 합니다.
- ☞ Sxx.S[0]이 On 되면 초기 스텝으로 복귀합니다.

알아두기

Step Controller 에서 Set 동작과 Jump 동작.

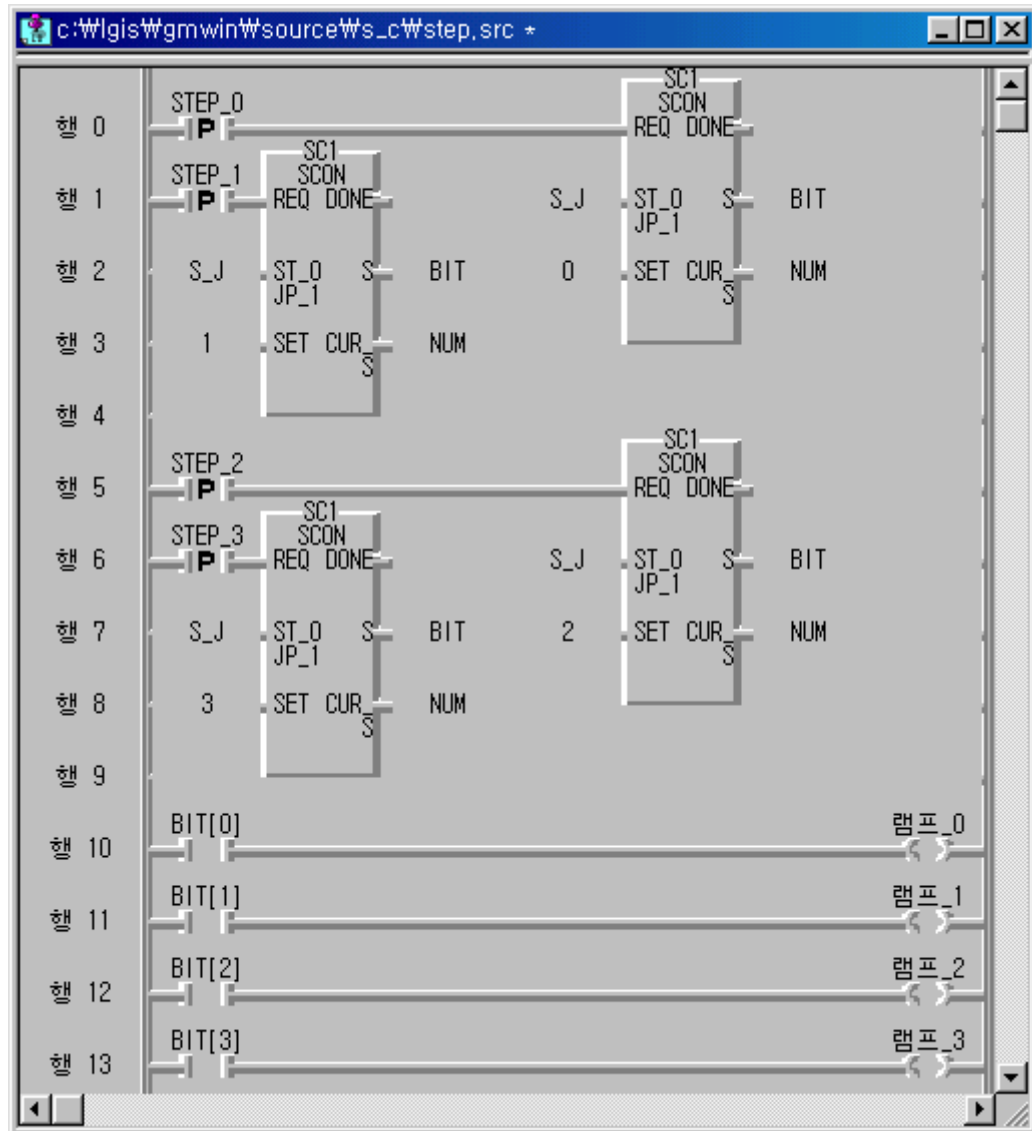
Step Controller 에서 Set 동작은 후진 및 점프 동작이 불가능합니다.

즉, Set 동작일 경우 1Step 씩 전진만 가능합니다.

Jump 동작은 전진, 후진, 점프가 가능하며, 여러 개의 입력 조건이 ON 되었을 경우 프로그램의 제일 끝 부분에 편집된 것을 우선으로 출력합니다.

○ 프로그램

S_J 스위치가 ON 되면 SET 동작을 하고 S_J 스위치가 OFF 되면 OUT 동작을 합니다.



	변수명	데이터타입	메모리할당	초기값	변수종류	사용여부	설명문
1	램프_0	BOOL	%QX0.1.0		VAR		
2	램프_1	BOOL	%QX0.1.1		VAR		
3	램프_2	BOOL	%QX0.1.2		VAR		
4	램프_3	BOOL	<자동>		VAR		
5	A3	BOOL	%QX0.1.3		VAR	*	
6	BIT	ARRAY[100] OF BOOL	<자동>		VAR	*	
7	NUM	INT	<자동>		VAR	*	
8	S_J	BOOL	%IX0.0.15		VAR	*	
9	SC1	FB Instance	<자동>		VAR	*	
10	STEP_0	BOOL	%IX0.0.0		VAR	*	
11	STEP_1	BOOL	%IX0.0.1		VAR	*	
12	STEP_2	BOOL	%IX0.0.2		VAR	*	
13	STEP_3	BOOL	%IX0.0.3		VAR	*	

부록 A. 표준 평선/평선블록 라이브러리

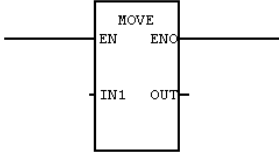
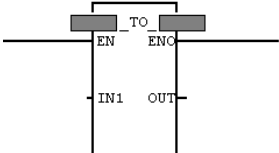
1. 표준 평선및 평선블록

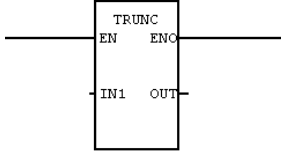
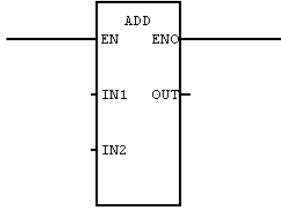
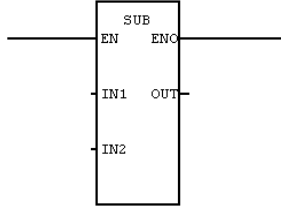
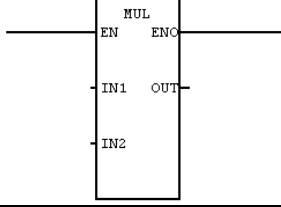
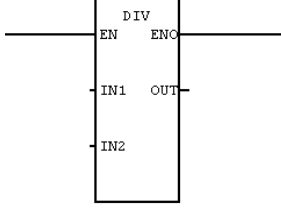
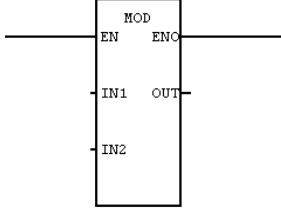
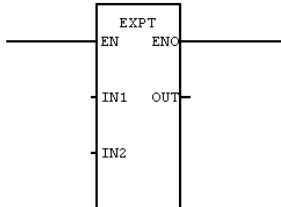
표준 평선에는 전송 평선, 변환 평선, 비교 평선, 산술연산평선, 논리연산 평선, 비트 시프트 평선, 선택 평선, 문자열 평선, 날짜 시각 평선, 시스템 제어 평선 등이 있고 표준 평선블록에는 카운터, 타이머, 에지검출, 바이스터블 평선블록 등이있습니다.

1.1 시퀀스 연산자

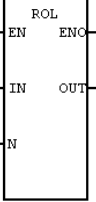
구분	명령어	기호	기능 설명	비 고
시 퀀 스 연 산 자	A 접점		A 접점 연산	
	B 접점		B 접점 연산	
	양변환 검출 접점		상승 에지에서 1Scan On 접점	
	음변환 검출 접점		하강 에지에서 1Scan On 접점	
	출력 코일		연산 결과 출력	
	반전 코일		연산 결과 반전 출력	
	출력 Set		연산 결과 세트 출력	
	출력 Reset		연산 결과 리셋 출력	
	양변환 검출 코일		연결선 연결될 때 1Scan On 출력	
	음변환 검출 코일		연결선 끊어질 때 1Scan On 출력	
	점프		레이블 위치로 점프	
	프로그램 종료		현재 프로그램 종료	

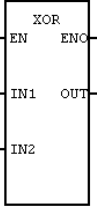
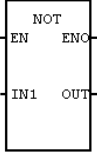


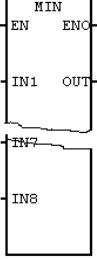
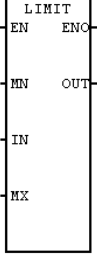
1.2 평선

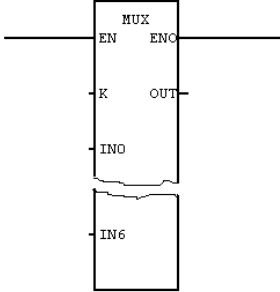
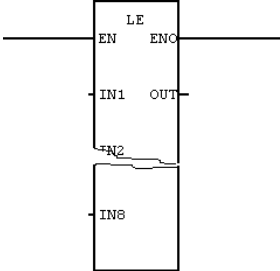
구분	명령어	기호	기능 설명	비 고
전 송 평 선	MOVE		<p>데이터 전송(복사)</p> <p>IN1 : MOVE 할 값(ANY)</p> <p>OUT : MOVE 된 값(ANY)</p>	IN, OUT 변수는 같은 Data Type 이어야 함.
변 환 평 선	****_TO_****		<p>데이터 형식 변환 평선</p> <p>IN1 : 전송원</p> <p>OUT : 전송선</p> <p>변환 명령 평선의 종류</p> <p>SINT_TO_INT 외 14 종</p> <p>INT_TO_SINT 외 14 종</p> <p>DINT_TO_SINT 외 14 종</p> <p>LINT_TO_SINT 외 14 종</p> <p>USINT_TO_SINT 외 14 종</p> <p>UINT_TO_SINT 외 14 종</p> <p>UDINT_TO_SINT 외 14 종</p> <p>ULINT_TO_SINT 외 14 종</p> <p>BYTE_TO_SINT 외 14 종</p> <p>WORD_TO_SINT 외 14 종</p> <p>DWORD_TO_SINT 외 14 종</p> <p>LWORD_TO_SINT 외 14 종</p> <p>BCD_TO_SINT 외 7 종</p> <p>REAL_TO_SINT 외 7 종</p> <p>LREAL_TO_SINT 외 7 종</p> <p>STRING_TO_SINT 외 18 종</p> <p>NUM_TO_STRING</p> <p>TIME_TO_UDINT 외 2 종</p> <p>DATE_TO_UINT 외 2 종</p> <p>TOD_TO_UDINT 외 2 종</p> <p>DT_TO_DATE 외 2 종</p>	<p>LINT</p> <p>ULINT</p> <p>LWORD</p> <p>REAL</p> <p>LREAL</p> <p>관련 평선은 GM1 과 GM2 만 가능</p>

구분	명령어	기호	기능 설명	비 고
변 환 평 선	TRUNC		<p>소수점 이하 값을 버리고 정수로 변환 (실수 > 정수)</p> <p>IN1 : 변환될 Real 값(ANY_REAL) OUT : 정수로 변환된 값(ANY_INT)</p>	GM1, GM2 전용
수 치 연 산 평 선	ADD		<p>덧셈</p> <p>IN1 ~ IN8 : 더할 값(ANY_NUM) OUT : 더한 결과값(ANY_NUM)</p>	IN, OUT 변수는 같은 Data Type 이어야 함.
	SUB		<p>뺄셈</p> <p>IN1 : 빼어질 값(ANY_NUM) IN2 : 뺄 값(ANY_NUM) OUT : 뺀 결과값(ANY_NUM)</p>	IN, OUT 변수는 같은 Data Type 이어야 함.
	MUL		<p>곱셈</p> <p>IN1 ~ IN8 : 곱할 값(ANY_NUM) OUT : 곱한 결과값(ANY_NUM)</p>	IN, OUT 변수는 같은 Data Type 이어야 함.
	DIV		<p>나눗셈(몫 구하기)</p> <p>IN1 : 나누어질 값(ANY_NUM) IN2 : 나눌 값(ANY_NUM) OUT : 몫(ANY_NUM) 단, 몫의 소수점 이하 버림.</p>	IN, OUT 변수는 같은 Data Type 이어야 함.
	MOD		<p>나눗셈(나머지 구하기)</p> <p>IN1 : 나누어질 값(ANY_INT) IN2 : 나눌 값(ANY_INT) OUT : 나머지 값(ANY_INT)</p>	IN, OUT 변수는 같은 Data Type 이어야 함.
	EXPT		<p>지수 연산</p> <p>IN1 : 진수(ANY_REAL) IN2 : 지수(ANY_NUM) OUT : 결과값(ANY_REAL) * GM1, GM2 전용</p>	IN1, OUT 변수는 같은 Data Type 이어야 함.

구분	명령어	기호	기능 설명	비 고
수 치 연 산 평 선	ABS		절대값 연산 IN1: 절대값연산 입력값(ANY_NUM) OUT: 절대값(ANY_NUM)	IN, OUT 은 모두 같은 Data Type 이어야 함.
	SQRT		제곱근 연산 IN1: 제곱근연산입력값(ANY_REAL) OUT: 제곱근값(ANY_REAL)	GM1, GM2 전용
	LN		자연대수 연산 IN1: 연산원(ANY_REAL) OUT: 자연대수값(ANY_REAL)	GM1, GM2 전용
	LOG		상용대수 연산 IN1: 연산원(ANY_REAL) OUT: 상용대수 연산값(ANY_REAL)	GM1, GM2 전용
	EXP		자연지수 연산 IN1: 연산원(ANY_REAL) OUT: 지수연산 결과값(ANY_REAL)	GM1, GM2 전용
삼 각 평 선	SIN		Sine 연산 IN1: 연산원(ANY_REAL) OUT: 결과값(ANY_REAL)	GM1, GM2 전용
	COS		Cosine 연산 IN1: 연산원(ANY_REAL) OUT: 결과값(ANY_REAL)	GM1, GM2 전용
	TAN		Tangent 연산 IN1: 연산원(ANY_REAL) OUT: 결과값(ANY_REAL)	GM1, GM2 전용
	ASIN		Arc Sine 연산 IN1: 연산원(ANY_REAL) OUT: 결과값(ANY_REAL)	GM1, GM2 전용
	ACOS		Arc Cosine 연산 IN1: 연산원(ANY_REAL) OUT: 결과값(ANY_REAL)	GM1, GM2 전용

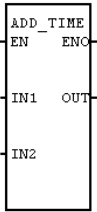

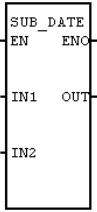
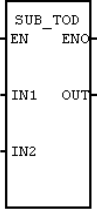
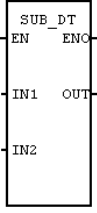
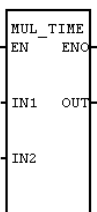
구분	명령어	기호	기능 설명	비 고
삼 각 평 선	ATAN		Arc Tangent 연산 IN1 : 연산원(ANY_REAL) OUT : 결과값(ANY_REAL)	GM1, GM2 전용
이 동 평 선	SHL		비트열 왼쪽으로 이동 IN : 이동될 비트열(ANY_BIT) N : 이동할 비트수(INT) OUT : 이동된 값(ANY_BIT)	
	SHR		비트열 오른쪽으로 이동 IN : 이동될 비트열(ANY_BIT) N : 이동할 비트수(INT) OUT : 이동된 값(ANY_BIT)	
회 전 명 령	ROL		비트열 왼쪽으로 회전 IN : 회전될 값(ANY_BIT) N : 회전할 비트수(INT) OUT : 회전된 값(ANY_BIT)	
	ROR		비트열 오른쪽으로 회전 IN : 회전될 값(ANY_BIT) N : 회전할 비트수(INT) OUT : 회전된 값(ANY_BIT)	
논 리 연 산	AND		논리곱 IN1 ~ IN8 : AND 될 값(ANY_BIT) OUT : AND 된 값(ANY_BIT)	IN, OUT 은 모두 같은 Data Type 이어야 함.
	OR		논리합 IN1 ~ IN8 : OR 될 값(ANY_BIT) OUT : OR 된 값(ANY_BIT)	IN, OUT 은 모두 같은 Data Type 이어야 함.

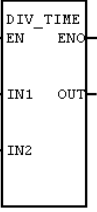
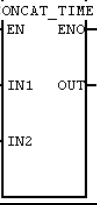
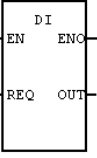
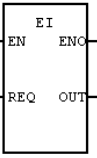
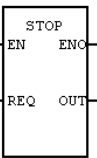
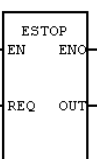
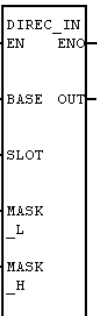
구분	명령어	기호	기능 설명	비 고
논 리 연 산	XOR		배타적 논리합(Exclusive OR) IN1 ~ IN8 : XOR 될 값(ANY_BIT) OUT : XOR 된 값(ANY_BIT)	IN, OUT 은 모두 같은 Data Type 이어야 함.
	NOT		논리 반전 IN1, IN2 : NOT 될 값(ANY_BIT) OUT : NOT 된 값(ANY_BIT)	IN, OUT 은 모두 같은 Data Type 이어야 함.
선 택 평 선	SEL		2 중 선택 G : 출력 선택(BOOL) > 0 또는 1 IN0 : G 가 0 일 경우 선택될 값 (Any) IN1 : G 가 1 일 경우 선택될 값 (Any) OUT : 선택된 값(Any)	IN, OUT 은 모두 같은 Data Type 이어야 함.
	MAX		최대값 구하기 IN1 ~ IN8 : 비교될 값(ANY_NUM) OUT : 입력값 중 최대값 (ANY_NUM)	IN, OUT 은 모두 같은 Data Type 이어야 함.
	MIN		최소값 구하기 IN1 ~ IN8 : 비교될 값(ANY_NUM) OUT : 입력값 중 최소값 (ANY_NUM)	IN, OUT 은 모두 같은 Data Type 이어야 함.
	LIMIT		상하한 제한값 MN : 최소값(ANY_NUM) IN : 제한될 값(ANY_NUM) MX : 최대값(ANY_NUM) OUT : 범위안에 든 값(ANY_NUM)	MIN, IN, MX, OUT 은 모두 같은 Data Type 이어야 함.


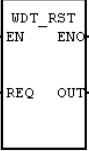
구분	명령어	기호	기능 설명	비 고
선택 평 선	MUX		<p>여러 개 중 선택(입력 최대 7 개)</p> <p>K : 선택 입력 번호(INT)</p> <p>IN0 : 전송원 0 번(Any)</p> <p>IN1 : 전송원 1 번(Any)</p> <p>IN2 : 전송원 2 번(Any)</p> <p>IN3 : 전송원 3 번(Any)</p> <p>IN4 : 전송원 4 번(Any)</p> <p>IN5 : 전송원 5 번(Any)</p> <p>IN6 : 전송원 6 번(Any)</p> <p>OUT : 선택된 값(Any)</p>	IN0, IN1, ... OUT 은 모두 같은 Data Type 이어야 함.
			<p>‘크다’ 비교</p> <p>IN1 ~ IN8 : 비교 데이터(Any)</p> <p>OUT : 비교 결과 값(BOOL)</p> <p>IN1 > IN2 > > IN7 > IN8 의 조건 성립시 OUT 출력 On</p>	IN1, IN2, ... 는 모두 같은 Data Type 이어야 함.
			<p>‘크거나 같다’ 비교</p> <p>IN1 ~ IN8 : 비교 데이터(Any)</p> <p>OUT : 비교 결과값(BOOL)</p> <p>IN1 ≥ IN2 ≥ ≥ IN7 ≥ IN8 의 조건 성립시 OUT 출력 On</p>	IN1, IN2, ... 는 모두 같은 Data Type 이어야 함.
			<p>‘같다’ 비교</p> <p>IN1 ~ IN8 : 비교 데이터(Any)</p> <p>OUT : 비교 결과값(BOOL)</p> <p>IN1 = IN2 = = IN7 = IN8 의 조건 성립시 OUT 출력 On</p>	IN1, IN2, ... 는 모두 같은 Data Type 이어야 함.
비 교 평 선	LE(≤)		<p>‘작거나 같다’ 비교</p> <p>IN1 ~ IN8 : 비교 데이터(Any)</p> <p>OUT : 비교 결과값(BOOL)</p> <p>IN1 ≤ IN2 ≤ ≤ IN7 ≤ IN8 의 조건 성립시 OUT 출력 On</p>	IN1, IN2, ... 는 모두 같은 Data Type 이어야 함.

구분	명령어	기호	기능 설명	비 고
	LT(<)		<p>‘작다’ 비교</p> <p>IN1 ~ IN8 : 비교 데이터(Any)</p> <p>OUT : 비교 결과값(BOOL)</p> <p>IN1 < IN2 < < IN7 < IN8 의 조건 성립시 OUT 출력 On</p>	IN1, IN2, ... 는 모두 같은 Data Type 이어야 함.
	NE(≠)		<p>‘같지 않다’ 비교</p> <p>IN1, IN2 : 비교 데이터(Any)</p> <p>OUT : 비교 결과값(BOOL)</p> <p>IN1 ≠ IN2 의 조건 성립시 OUT 출력 On</p>	IN1, IN2 는 모두 같은 Data Type 이어야 함.
문 자 열 평 선	LEN		<p>문자열 길이</p> <p>IN1 : 문자열 입력(String)</p> <p>OUT : 문자열 길이(INT)</p>	
	LEFT		<p>문자열 왼쪽 부분 전송</p> <p>IN : 문자열 입력(String)</p> <p>L : 문자열 길이(INT)</p> <p>OUT : 문자열 출력(String)</p>	
	RIGHT		<p>문자열 오른쪽 부분 전송</p> <p>IN : 문자열 입력(String)</p> <p>L : 문자열 길이(INT)</p> <p>OUT : 문자열 출력(String)</p>	
	MID		<p>문자열 중간 부분 전송</p> <p>IN : 문자열 입력(String)</p> <p>L : 문자열 길이(INT)</p> <p>P : 문자열 선두 위치(INT)</p> <p>OUT : 문자열 출력(String)</p>	

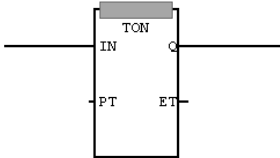
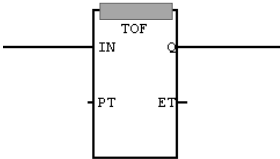
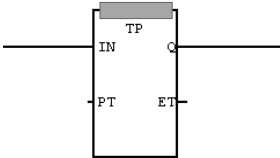
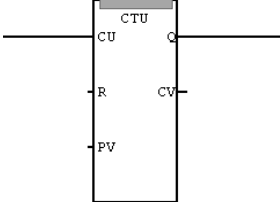
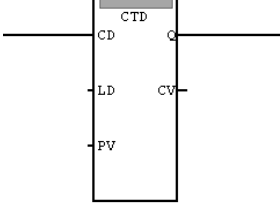
구분	명령어	기호	기능 설명	비 고
문 자 열 평 선	CONCAT		<p>문자열 연결</p> <p>입력 문자열을 순서대로 연결</p> <p>IN1 ~ IN8 : 문자열(String)</p> <p>OUT : 문자열 출력(String)</p>	
	INSERT		<p>문자열 삽입</p> <p>IN1 : 문자열 입력(String)</p> <p>IN2 : 삽입할 문자열(String)</p> <p>P : 문자열 선두 위치(INT)</p> <p>OUT : 문자열 출력(String)</p>	
	DELETE		<p>문자열 삭제</p> <p>IN1 : 문자열 입력(String)</p> <p>L : 삭제할 문자열 길이(INT)</p> <p>P : 문자열 선두 위치(INT)</p> <p>OUT : 문자열 출력(String)</p>	
	REPLACE		<p>문자열 대체</p> <p>IN1 : 문자열 입력(String)</p> <p>IN2 : 대체할 문자열(String)</p> <p>P : 문자열 선두 위치(INT)</p> <p>OUT : 문자열 출력(String)</p>	
	FIND		<p>문자열 찾기</p> <p>IN1 : 문자열 입력(String)</p> <p>IN2 : 검색할 문자열(String)</p> <p>OUT : 문자열 선두 위치(INT)</p>	

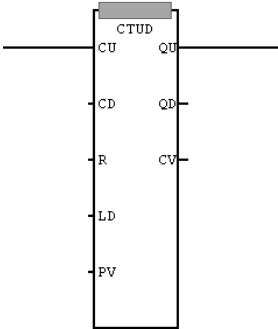
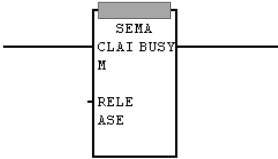
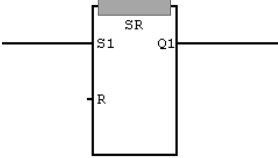
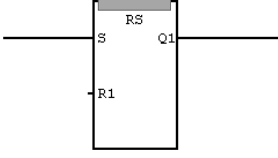
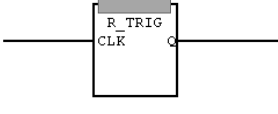
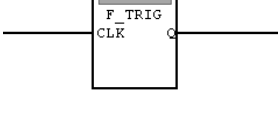
구분	명령어	기호	기능 설명	비 고
날 짜 시 간 평 선	ADD_TIME		<p>시간 더하기</p> <p>IN1 : 시각 또는 시간 (TIME, TOD, TD)</p> <p>IN2 : 더할 시간(TIME)</p> <p>OUT : 결과 시각 또는 시간 (TIME, TOD, TD)</p>	
	SUB_TIME		<p>시간 빼기</p> <p>IN1 : 시각 또는 시간 (TIME, TOD, TD)</p> <p>IN2 : 뺄 시간(TIME)</p> <p>OUT : 결과 시각 또는 시간 (TIME, TOD, TD)</p>	
	SUB_DATE		<p>날짜 빼기</p> <p>IN1 : 날짜(DATE)</p> <p>IN2 : 뺄 날짜(DATE)</p> <p>OUT : 결과 시간(TIME)</p>	
	SUB_TOD		<p>시각 빼기</p> <p>IN1 : 시각(TIME OF DAY)</p> <p>IN2 : 뺄 시각(TIME OF DAY)</p> <p>OUT : 결과 시간(TIME)</p>	
	SUB_DT		<p>날짜 시각 빼기</p> <p>IN1 : 시각(DATE&TIME)</p> <p>IN2 : 뺄 시각(DATE&TIME)</p> <p>OUT : 결과 시간(TIME)</p>	
	MUL_TIME		<p>시간 곱하기</p> <p>IN1 : 입력 시간(TIME)</p> <p>IN2 : 곱할 값(INT)</p> <p>OUT : 결과 시간(TIME)</p>	

구분	명령어	기호	기능 설명	비 고
날 짜 시 간 평 선	DIV_TIME		<p>시간 나누기</p> <p>IN1 : 입력 시간(TIME)</p> <p>IN2 : 나눌 값(INT)</p> <p>OUT : 결과 시간(TIME)</p>	
	CONCAT_TIME		<p>날짜와 시각 연결</p> <p>IN1 : 입력 날짜(DATE)</p> <p>IN2 : 입력 시각(TOD)</p> <p>OUT : 결과 날짜 시각(DT)</p>	
시 스 템 제 어 평 선	DI		<p>인터럽트 금지</p> <p>REQ : 1로 금지 요구(BOOL)</p> <p>OUT : 금지 확인으로 1 출력(BOOL)</p>	
	EI		<p>인터럽트 허가</p> <p>REQ : 1로 허가 요구(BOOL)</p> <p>OUT : 허가 확인으로 1 출력(BOOL)</p>	
	STOP		<p>PLC 프로그램 정지 요구</p> <p>REQ : 1로 정지 요구(BOOL)</p> <p>OUT : 정지 확인으로 1 출력(BOOL)</p>	스캔 완료 후 프로그램 중지
	ESTOP		<p>PLC 프로그램 비상 정지 요구</p> <p>REQ : 1로 정지 요구(BOOL)</p> <p>OUT : 정지 확인으로 1 출력(BOOL)</p>	즉시 프로그램 중지
	DIREC_IN		<p>입력 데이터 즉시 갱신(Refresh)</p> <p>BASE: 입력모듈 장착된 베이스 번호</p> <p>SLOT: 입력모듈 장착된 슬롯 번호</p> <p>MASK_L : 하위 32Bit 중 갱신 하지 않을 Bit 지정(DWORD)</p> <p>MASK_H : 상위 32Bit 중 갱신 하지 않을 Bit 지정(DWORD)</p> <p>OUT: 입력데이터 갱신이 완료되면 1 출력(BOOL)</p>	

구분	명령어	기호	기능 설명	비 고
시스템 제어 평선	DIREC_O		출력 데이터 즉시 갱신(Refresh) BASE: 출력모듈 장착된 베이스 번호 SLOT: 출력모듈 장착된 슬롯 번호 MASK_L : 하위 32Bit 중 갱신 하지 않을 Bit 지정(DWORD) MASK_H : 상위 32Bit 중 갱신 하지 않을 Bit 지정(DWORD) OUT: 출력데이터 갱신이 완료되면 1 출력(BOOL)	
	WDT_RST		워치 독(Watch Dog) 타이머 리셋 REQ: 워치 독 타이머 리셋 요구(BOOL) OUT: 워치 독 타이머 초기화 후 1 출력(BOOL)	1 스캔 중 중복 사용 가능

1.3 평선 블록

구분	명령어	기호	기능 설명	비 고
타 이 머 평 선 블 록	TON		On 딜레이 타이머 IN : 타이머 기동 조건(BOOL) PT : 설정 시간(TIME) Q : 타이머 출력(BOOL) ET : 경과 시간	IN 이 ON 된 후, PT 에 도달하면 타이머 출력. 단, IN 이 OFF 면 타이머 출력 즉시 OFF
	TOF		Off 딜레이 타이머 IN : 타이머 기동 조건(BOOL) PT : 설정 시간(TIME) Q : 타이머 출력(BOOL) ET : 경과 시간	IN 이 ON 되면 타이머 출력. 단, IN 이 OFF 면 PT 경과 후 타이머 출력 OFF
	TP		펄스 타이머 IN : 타이머 기동 조건(BOOL) PT : 설정 시간(TIME) Q : 타이머 출력(BOOL) ET : 경과 시간(TIME)	IN 이 ON 되면 PT 동안만 타이머 출력
카 운 터 평 선 블 록	CTU		Up 카운터 CU : 가산 펄스 입력(BOOL) R : 리셋 입력(BOOL) PV : 설정값(INT) Q : 카운터 출력(BOOL) CV : 현재값(INT)	현재값이 설정값 이상이면, 카운터 출력 ON
	CTD		Down 카운터 CD : 감산 펄스 입력(BOOL) LD : 설정값 입력 LD 가 1 이면 PV 를 CV 로 로드(BOOL) PV : 설정값(INT) Q : 카운터 출력(BOOL) CV : 현재값(INT)	현재값이 0 이하면 카운터 출력 ON

구분	명령어	기호	기능 설명	비 고
카 운 터 평 선 블 록	CTUD		가감산(Up/Down) 카운터 CU : 가산 펄스 입력(BOOL) CD : 감산 펄스 입력(BOOL) R : 리셋 입력(BOOL) LD : 로드 입력(BOOL) PV : 설정값(INT) QU : 업 카운터 출력(BOOL) QD : 다운 카운터 출력(BOOL) CV : 현재값(INT)	QU ON 조건 : CV 가 PV 이상일 때 QD ON 조건 : CV 가 0 이하일 때
평 선 블 록	SEMA		시스템 자원 제어(Semaphore) CLAIM : 자원 독점 요구신호 (BOOL) RELEASE : 해제 신호(BOOL) BUSY : 자원 취득불가신호(대기) (BOOL)	
	SR		Set 우선 쌍안정(Bistable) S1 : Set 조건(BOOL) R : Reset 조건(BOOL) Q1 : 연산결과(BOOL)	
	RS		Reset 우선 쌍안정(Bistable) S : Set 조건(BOOL) R1 : Reset 조건(BOOL) Q1 : 연산결과(BOOL)	
	R_TRIG		상승 에지 검출 CLK : 입력(BOOL) Q : 출력(BOOL)	
	F_TRIG		하강 에지 검출 CLK : 입력(BOOL) Q : 출력(BOOL)	

1 기본 평션 일람

전송 평션

평션	기능	비고
MOVE	입력 변수를 출력 변수로 복사	
ARY_MOVE	입력 배열 변수의 원소를 출력 배열 변수로 복사	

산술 연산 평션

평션	기능	비고
ABS	절대값 연산	
ACOS	Arc Cosine 연산	GMR1/2
ADD	덧셈 연산	
ASIN	Arc Sine 연산	GMR1/2
ATAN	Arc Tangent 연산	GMR1/2
COS	Cosine 연산	GMR1/2
DIV	나눗셈 연산(몫 구하기)	
EXP	자연 지수 연산($OUT = e^{IN}$)	GMR1/2
EXPT	지수 연산($OUT = IN1^{IN2}$)	GMR1/2
LN	자연 대수 연산($OUT = \ln IN$)	GMR1/2
LOG	상용 대수 연산($OUT = \log_{10} IN = \log IN$)	GMR1/2
MOD	나머지 구하기 연산	
MUL	곱셈 연산	
SIN	Sine 연산	GMR1/2
SQRT	제곱근 연산($OUT = \sqrt{IN}$)	GMR1/2
SUB	뺄셈 연산	
TAN	Tangent 연산	GMR1/2

1.3 시스템 평션

평션	기능	비고
DI	태스크 프로그램 기동 불허	
DIRECT_IN	입력 데이터 즉시 갱신	
DIRECT_O	출력 모듈 데이터 즉시 갱신	
EI	태스크 프로그램 기동 허가(DI 의 해제)	
ESTOP	프로그램에 의한 비상 운전정지	
STOP	프로그램에 의한 운전정지	
WDT_RST	Watch_Dog Timer 초기화	

1.4 시간 연산 평션

평 선행	기 능	비 고
ADD_TIME	시간의 덧셈	
CONTACT_TIME	IN1(날짜)과 IN2(시각)를 붙여서 날짜와 시각 (DATE_AND_TIME) OUT 으로 출력	
DIV_TIME	IN1(시간)을 IN2(숫자)로 나누어서 나누어진 시간을 OUT 으로 출력	
MUL_TIME	IN1(시간)을 IN2(숫자)로 곱해서 결과 시간을 OUT 으로 출력	
SUB_DATE	IN1(기준날짜)에서 IN2(특정날짜)를 빼서 날짜 차이를 OUT	
SUB_DT	IN1(기준 날짜와 시각)에서 IN2(특정 날짜와 시각)를 빼서 시간 차이를 OUT 으로 출력	
SUB_TIME	시간의 뺄셈	
SUB_TOD	IN1(기준시각)에서 IN2(특정시각)를 빼서 시간 차이를 OUT 으로 출력	

1.5 타입 변환 평션

평 선행	기 능	비 고
ARY_TO_STRING	Byte Array 를 문자열로 변환	
BCD_TO_***	BCD 타입을 *** 타입으로 변환	
BOOL_TO_***	BOOL 타입을 *** 타입으로 변환	
BYTE_TO_***	BYTE 타입을 *** 타입으로 변환	
DATE_TO_***	DATE 타입을 *** 타입으로 변환	
DINT_TO_***	DINT 타입을 *** 타입으로 변환	
DT_TO_***	DT 타입을 *** 타입으로 변환	
DWORD_TO_***	DWORD 타입을 *** 타입으로 변환	
INT_TO_***	INT 타입을 *** 타입으로 변환	
LINT_TO_***	LINT 타입을 *** 타입으로 변환	GMR/1/2
LREAL_TO_***	LREAL 타입을 *** 타입으로 변환	GMR/1/2
LWORD_TO_***	LWORD 타입을 *** 타입으로 변환	GMR/1/2
NUM_TO_STRING	숫자 데이터를 문자 데이터로 변환	
REAL_TO_***	REAL 타입을 *** 타입으로 변환	GMR/1/2
SINT_TO_***	SINT 타입을 *** 타입으로 변환	
STRING_TO_***	STRING 타입을 *** 타입으로 변환	
STRING_TO_ARY	String 을 Byte Array 로 변환(최대 30 문자)	
TIME_TO_***	TIME 타입을 *** 타입으로 변환	
TOD_TO_***	TOD 타입을 *** 타입으로 변환	
UDINT_TO_***	UDINT 타입을 *** 타입으로 변환	
UNIT_TO_***	UINT 타입을 *** 타입으로 변환	
ULINT_TO_***	UL INT 타입을 *** 타입으로 변환	GMR/1/2
USINT_TO_***	USINT 타입을 *** 타입으로 변환	
WORD_TO_***	WORD 타입을 *** 타입으로 변환	

1.6 비트열 연산 평션

평션	기능	비고
AND	IN1을 IN2와 비트별로 AND 해서 OUT으로 출력	
NOT	IN을 비트별로 반전해서 OUT으로 출력	
OR	IN1을 IN2와 비트별로 OR 해서 OUT으로 출력	
ROL	IN을 설정한 비트 수(N) 만큼 왼쪽으로 회전	
ROR	IN을 설정한 비트 수(N) 만큼 오른쪽으로 회전	
SHL	IN을 설정한 비트 수(N) 만큼 왼쪽으로 이동 OUT의 오른쪽 N개 비트는 0으로 채움	
SHR	IN을 설정한 비트 수(N) 만큼 오른쪽으로 이동 OUT의 왼쪽 N개 비트는 0으로 채움	
XOR	IN1을 IN2와 비트별로 XOR 해서 OUT으로 출력	

1.7 문자열 연산 평션

평션	기능	비고
CONTACT	문자열 연결하기	
DELETE	문자열에서 특정 부분 삭제	
FIND	입력 문자열 IN1에서 문자열 IN2의 위치 찾기	
INSERT	문자열 삽입하기	
LEFT	입력 문자열의 왼쪽부터 설정된 길이만큼 출력	
LEN	입력 문자열의 문자 수 출력	
MID	입력 문자열의 지정된 위치의 출력	
REPLACE	입력 문자열의 특정 부분을 다른 문자로 대체하기	
RIGHT	입력 문자열의 오른쪽부터 설정된 길이만큼 출력	

1.8 비교/선택 평션

평션	기능	비고
EQ	$IN1=IN2=IN3\ldots=INn$ 이면 OUT으로 1 출력($N \leq 8$)	
GE	$IN1 \geq IN2 \geq IN3 \ldots \geq INn$ 이면 OUT으로 1 출력($N \leq 8$)	
GT	$IN1 > IN2 > IN3 \ldots > INn$ 이면 OUT으로 1 출력($N \leq 8$)	
LE	$IN1 \leq IN2 \leq IN3 \ldots \leq INn$ 이면 OUT으로 1 출력($N \leq 8$)	
LIMIT	$MN \leq IN \leq MX$ 이면 OUT으로 IN 출력	
LT	$IN1 < IN2 < IN3 \ldots < INn$ 이면 OUT으로 1 출력($N \leq 8$)	
MAX	IN1, IN2, ..., INn 중에서 최대값을 OUT으로 출력($N \leq 8$)	
MIN	IN1, IN2, ..., INn 중에서 최소값을 OUT으로 출력($N \leq 8$)	
MUX	IN0, IN1, ..., INn 중 1개를 선택하여 출력($N \leq 6$)	
NE	2개의 입력이 같지 않으면 OUT으로 1 출력	
SEL	2개의 입력 중 1개를 선택	

2. 응용 평션 라이브러리

2.1 MASTER-K 평션 라이브러리(MKSTDLIB.xFU)

평션	기능	비고
BMOV_***	*** 크기의 비트열의 일부분을 복사,이동	출력 데이터 타입이 L 인 경우 GMR/1/2 에서만 사용가능
BSUM_***	*** 크기의 비트열에서 ON 된 비트수를 숫자로 출력	
DEC_***	*** 크기의 비트열 데이터를 1 만큼 감소시켜 출력	
DECO_***	*** 크기의 비트열 출력 데이터 중 지정된 위치의 비트만 1로 하여 출력	
ENCO_***	*** 크기의 입력 비트열 데이터 중, 1로 되어있는 비트 중 최상위 비트의 위치를 OUT으로 출력	
INC_***	*** 크기의 비트열 데이터를 1 만큼 증가시켜 출력	
SEG	BCD 또는 HEX 값을 7 세그먼트 코드로 변환	

2.2 응용 평션 라이브러리(APP.xFU)

2.2.1 배열 변수 평션

평션	기능	비고
ARY_ASC_TO_BCD	ASCII 데이터의 WORD Array 를 BCD 값의 BYTE Array 로 변환	
ARY_ASC_TO_BYTE	ASCII 데이터의 WORD Array 를 HEX 값의 BYTE Array 로 변환	
ARY_AVE_***	*** 타입의 Array 변수 데이터의 평균	
ARY_BCD_TO_ASC	BCD 값의 BYTE Array 를 ASCII 데이터의 WORD Array 로 변환	
ARY_BYTE_TO_ASC	HEX 값의 BYTE Array 를 ASCII 데이터의 WORD Array 로 변환	
ARY_CMP	*** 타입의 두 Array 변수의 데이터 비교	
ARY_FLL_***	입력값으로 Array 의 데이터 변경	
ARY_MOVE	입력 Array 의 데이터를 출력 Array 로 복사	
ARY_ROT_C_***	지정된 범위의 Array 원소들의 bit 들을 정해진 비트 수만큼 회전	
ARY_SCH_***	Array 내에서 입력된 값과 동일한 값을 찾아 Array 내에서의 처음 위치와 전체 개수를 출력	
ARY_SFT_C_***	Array 원소들의 bit 들을 정해진 개수만큼 지정된 방향으로 이동	
ARY_SWAP_***	입력된 Array 원소를 2 개의 크기로 구분하여 상위와 하위를 서로 교환	
ROTATE_A_***	Array 블록 중 지정된 범위의 원소들을 지정된 방향으로 회전	
SHIFT_A_***	Array 블록 중 지정된 범위의 원소들을 지정된 방향으로 이동	

2.2.2 타입 변환 평션

평션	기능	비고
ASC_TO_BCD	2 개의 ASCII 값을 입력 받아 2 자리의 BCD 로 출력	
ASC_TO_BYTE	2 개의 ASCII 값을 입력 받아 2 자리의 HEX 로 출력	
BCD_TO_ASC	2 자리의 BCD 값을 입력 받아 2 개의 ASCII 값 출력	
BIT_BYTE	8 개의 비트를 1 개의 바이트로 조합	
BYTE_BIT	1 개의 바이트를 8 개의 비트로 분산	
BYTE_TO_ASC	2 자리의 HEX 값을 입력 받아 2 개의 ASCII 값 출력	
BYTE_WORD	2 개의 바이트를 1 개의 워드로 조합	
DEG_***	Radian 값을 입력 받아 각도(Degree)로 출력	R/1/2
DWORD_LWORD	2 개의 DWORD 를 1 개의 LWORD 로 조합	R/1/2
DWORD_WORD	1 개의 DWORD 를 2 개의 WORD 로 분산	
LWORD_DWORD	1 개의 LWORD 를 2 개의 DWORD 로 분산	R/1/2
RAD_***	각도(°)를 입력 받아 Radian 값으로 출력	R/1/2
WORD_BYTE	하나의 워드를 2 개의 바이트로 분산	
WORD_DWORD	2 개의 WORD 를 하나의 DWORD 로 조합	

2.2.3 기타 응용 평션

평션	기능	비고
DIS_***	입력 데이터를 지정된 비트 개수 단위로 구분	
GET_CHAR	STRING 의 지정된 위치로부터 1 개의 바이트를 추출	
MCS	Master Control	
MCSCCLR	Master Control 해제	
MEQ	입력된 2 개의 데이터에 Masking 후 데이터 비교	
PUT_CHAR	1 개의 바이트 입력값을 STRING 상의 지정된 위치에 덮어쓰기	
ROTATE_C_***	비트 열 중 지정된 bit 들을 지정된 방향으로 회전	
RTC_SET	설정된 DATA 를 PLC 의 Clock Device 에 저장	
SHIFT_C_***	비트 열 중 지정된 bit 들을 지정된 방향으로 이동	
SWAP_***	입력된 변수를 2 개의 크기로 구분하여 상위와 하위를 서로 교환	
UNI_***	입력 Array 를 지정한 비트 수 별로 하위 비트부터 지정된 비트 수 만큼 결합	
XCHG	2 개의 입력 데이터를 교환	

3 기본 펄션 블록 라이브러리

3.1 카운터 및 타이머

펄션 블록	기 능	비고
CTD	다운 카운터 (현재값 : -32768 ~ 32767)	
CTU	업 카운터 (현재값 : -32768 ~ 32767)	
CTUD	업 다운 카운터 (현재값 : -32768 ~ 32767)	
TOF	OFF Delay 타이머	타임형 설정값(T#)
TON	ON Delay 타이머	
TP	펄스 타이머	

3.2 기타 기본 펄션 블록

펄션 블록	기 능	비고
F_TRIG	하강 에지 검출	
RS	Reset 우선 Bistable	
R_TRIG	상승 에지 검출	
SEMA	시스템 자원에 대한 독점적 제어권을 취득	
SR	Set 우선 Bistable	

4 응용 펄션 블록 라이브러리(APP.xFB)

펄션 블록	기 능	비고
CTR	링 카운터	
DUTY	스캔 지정 ON/OFF	
FIFO_***	FIFO 스택에 값을 Load/Unload (선입 선출)	
LIFO_***	LIFO 스택에 값을 Load/Unload (후입 선출)	
SCON	Step Controller	
TMR	적산 타이머	
TMR_FLK	플리커 타이머	
TMR_UINT	정수 설정 적산 타이머	
TOF_RST	동작 중 출력 OFF 가 가능한 OFF Delay 타이머	
TOF_UINT	정수 설정 OFF Delay 타이머	
TON_UINT	정수 설정 ON Delay 타이머	
TP_RST	동작 중 출력 OFF 가 가능한 펄스 타이머	
TP_UINT	정수 설정 펄스 타이머	
TRTG	리트리거블 타이머	
TRTG_UINT	정수 설정 리트리거블 타이머	

1. 수치체계 및 데이터구조

1.1 수치(데이터)의 표현

PLC CPU에서는 모든 정보를 On 과 Off, 또는 “1”과 “0”의 상태로 기억하고 처리합니다. 따라서 수치 연산도 1 과 0 으로 처리된 수치, 즉 2 진수 (Binary number BIN)로 처리합니다.

한편, 일상 생활에서는 10 진수가 알기 쉽고 가장 널리 사용되고 있습니다. 그래서 PLC 에 수치를 Write 할 경우, 또는 PLC 의 수치정보를 Read 할 경우에는 10 진수에서 16 진수로, 16 진수에서 10 진수로 변환이 필요합니다

1.1.1 10 진수(Decimal)

10 진수란 “ 0~9 의 종류의 기호를 사용하여 순서와 크기(량)를 표현하는 수”를 말합니다.

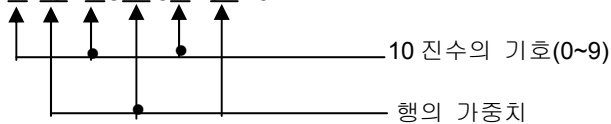
그리고 0, 1, 2, 3, 4,9 다음에 “10”으로 자리올림하고 계속 진행됩니다.

예를 들면, 10 진수「153」을 행과 “행의 가중치”란 측면에서 보면 아래와 같습니다.

$$153=100+50+3$$

$$=1\times 100+5\times 10+3\times 1$$

$$=1\times 10^2 + 5\times 10^1 + 3\times 10^0$$



1.1.2 2 진수 (Binary ... Bin)

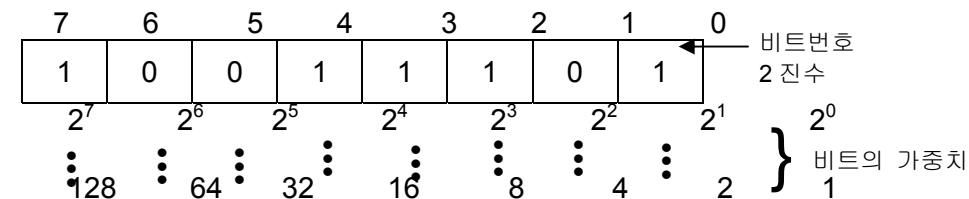
2 진수란 “0 과 1 의 두 종류 기호를 사용하여 순서와 크기를 나타내는 수”를 말합니다. 그래서 0, 1 다음에 “10”으로 자리올림을 하고, 계속 진행됩니다. 즉, 0,1 의 한 자리 수를 비트라고 합니다.

2 진수	10 진수
0	0
1	1
10	2
11	3
100	4
101	5

예를 들면 다음의 2 진수는 10 진수로 얼마나 되는지 생각해 봅시다.

“10011101”

10 진수에서 행번호와 행의 가중치를 고려하였듯이 우측부터 비트번호와 비트가중치를 붙여 봅시다.



10 진수와 같이 각 비트의 코드의 가중치의 곱의 합을 생각해 봅시다.

$$\begin{aligned}
 &= 1 \times 128 + 0 \times 64 + 0 \times 32 + 1 \times 16 + 1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1 \\
 &= 128 + 16 + 8 + 4 + 1 \\
 &= 157
 \end{aligned}$$

즉, 2 진수는 “코드가 1 인, 비트의 가중치를 가산한 것”이 10 진수로 되는 것입니다.

일반적으로 8 비트를 1 바이트, 16 비트 (2 바이트)를 1 워드라 말합니다.



1.1.3 16 진수 (Hexadecimal HEX)

16 진수도 10 진수, 2 진수와 동일하게 생각하여 “0 ~ 9, A ~ F 의 종류의 기호를 사용하여 순서와 크기를 나타내는 수”를 말합니다.

그리고 0, 1, 2,D,E,F 다음에 “10”으로 자리올림을 하고 계속 진행됩니다.

10 진수	16 진수	2 진수
4	4	100
5	5	101
6	6	110
7	7	111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111
16	10	10000
17	11	10001
18	12	10010
.	.	.
.	.	.
.	.	.

1 9 1 0 1 = 4 A 9 D = 0100 1010 1001 1101

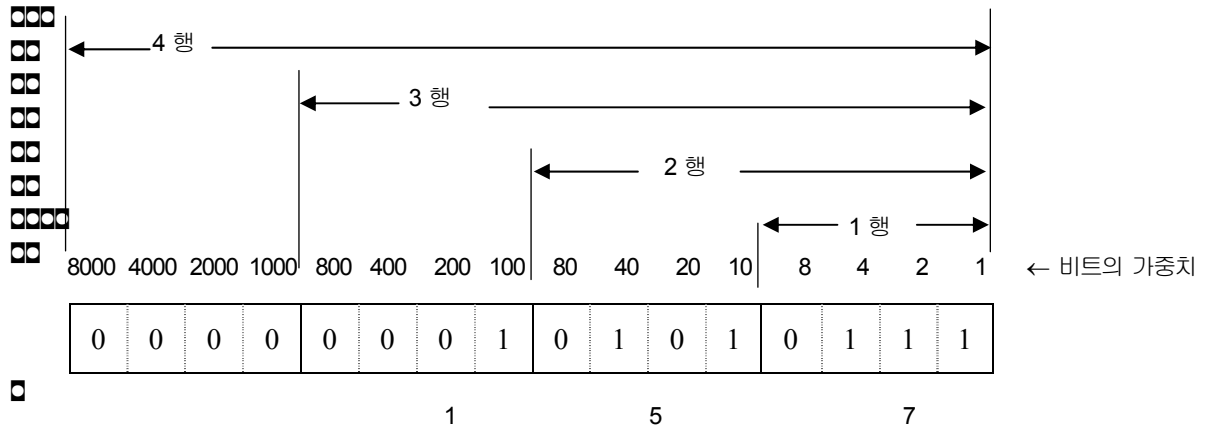
3	2	1	0	← 행번
4	A	9	D	← 16 진수

$$\begin{aligned}
 &= (4) \times 16^3 + (A) \times 16^2 + (9) \times 16^1 + (D) \times 16^0 \\
 &= 4 \times 4096 + 10 \times 2568 + 9 \times 16 + 13 \times 1 \\
 &= \underline{19101}
 \end{aligned}$$

16 진수의 한자리는 2 진수의 4 비트로 대응됩니다.

1.1.4 2 진화 10 진수 (Binary Coded Decimal (BCD))

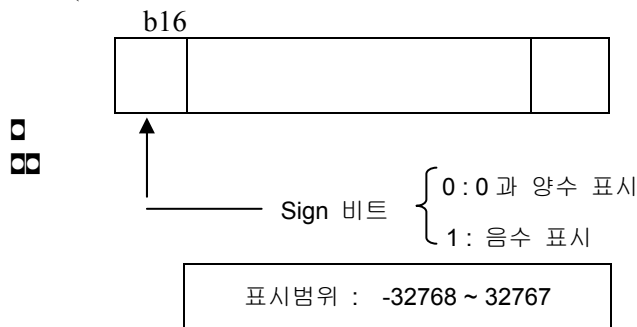
2 진화 10 진수는 “10 진수의 각행의 숫자를 2 진수로 나타낸 수”를 말합니다.
 예를들면, 10 진수의 157 는 다음과 같이 10 진수의 0 ~ 9999 (4 행의 최대치)를
 16 비트로 나타냅니다.
 각 비트의 가중치는 다음과 같습니다.



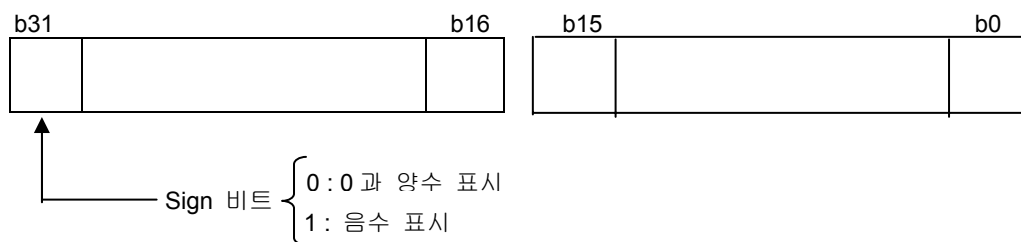
1.2 정수의 표현

정수 표시는 최상위 비트(MSB)가 0 이되면 양수를 나타내고 1 이면 음수로 나타나게 됩니다.
 이때 0, 1 에 따라 음수 양수를 표시하는 최상위 비트를 Sign 비트라고 합니다.
 16 비트도 32 비트에서는 MSB 의 위치가 다르기 때문에 Sign 비트 위치에 주의해야 합니다.

(16 비트 일 경우



* 32 비트 일 경우



표시범위 : -2147483648 ~ 2147483647

1.3 음수의 표현

음수를 바이너리 코드로 표현하고자 할 때는 먼저 부호를 생략한 데이터를 바이너리 코드로 전환한 후 1의 보수를 취하면 됩니다.

예) -0001을 표기하는 방법

(1) 부호를 생략한 0001을 바이너리 코드로 전환한다. (b0=1)

b15			b0	
1	0	~	0	1

(2) (1)의 결과를 반전시킨다. (b0 = 0)

b15			b0	
1	1	~	1	0

(3) (2)의 결과에 +1을 한다.

b15					b0	
1	1	1	1	1	~	1
1						1

-0001 = hFFFF

즉, 음의 정수 -0001을 바이너리 코드로 전환했을 경우 hFFFF와 동일한 결과가 나타납니다. 또 무부호 정수(UINT) 정수의 최대값 65,535를 바이너리 코드로 전환해 보면 hFFFF가 나타납니다.

역으로 말하면, hFFFF는 -0001 또는 65,535로 인식할 수 있는데, 변수 설정 시 설정한 데이터 타입에 따라 정수(INT) 또는 무부호 정수(UINT)로 인식하게 됩니다.

부록 D. 플래그 일람표 (예약 변수)

1. 예약 변수

- ▷ 예약 변수는 시스템에서 미리 선언한 변수들입니다. 이 변수들은 특수한 용도로 사용하며, 사용자가 이 변수 이름으로 변수 선언을 할 수는 없습니다.
- ▷ 이 예약 변수를 사용할 때에는 변수 선언 없이 사용합니다.

1) 사용자 플래그

예 약 변 수	데 이 터 타 입	내 용
ERR	BOOL	연산 에러 접점
LER	BOOL	연산 에러 래치 접점
T20MS	BOOL	20 ms 클럭 접점
T100MS	BOOL	100 ms 클럭 접점
T200MS	BOOL	200 ms 클럭 접점
T1S	BOOL	1 초 클럭 접점
T2S	BOOL	2 초 클럭 접점
T10S	BOOL	10 초 클럭 접점
T20S	BOOL	20 초 클럭 접점
T60S	BOOL	60 초 클럭 접점
ON	BOOL	항시 On 접점
OFF	BOOL	항시 Off 접점
1ON	BOOL	1 스캔 On 접점
1OFF	BOOL	1 스캔 Off 접점
STOG	BOOL	스캔마다 반전
INIT_DONE	BOOL	초기화 프로그램 완료
RTC_DATE	DATE	RTC의 현재 날짜
RTC_TOD	TOD	RTC의 현재 시간
RTC_WEEK	UINT	RTC의 현재 요일

2) 시스템 에러 대표 플래그

예 약 변 수	데 이 터 타 입	내 용
CNF_ER	WORD	시스템의 에러(중고장)
CPU_ER	BOOL	CPU 구성 에러
IO_TYER	BOOL	모듈 타입 불일치 에러
IO_DEER	BOOL	모듈 착탈 에러
FUSE_ER	BOOL	Fuse 단선 에러
IO_RWER	BOOL	입출력 모듈 읽기/쓰기 에러(고장)
SP_IFER	BOOL	특수/통신 모듈 인터페이스 에러(고장)
ANNUN_ER	BOOL	외부기기의 중고장 검출 에러
WD_ER	BOOL	Scan Watch-Dog 에러
CODE_ER	BOOL	프로그램 코드 에러
STACK_ER	BOOL	Stack Overflow 에러
P_BCK_ER	BOOL	프로그램 에러

3) 시스템 에러 해제 플래그

예 약 변 수	데 이 터 타 입	내 용
CNF_ER_M	BYTE	시스템 에러(중고장) 해제
IO_DEER_M	BOOL	모듈 착탈 에러 해제
FUSE_ER_M	BOOL	퓨즈 단선 에러 해제
IO_RWER_M	BOOL	입출력 모듈 읽기/쓰기 에러 해제
SP_IFER_M	BOOL	특수/통신 모듈 인터페이스 에러 해제
ANNUN_ER_M	BOOL	외부기기의 중고장 검출 에러 해제

4) 시스템 경고 대표 플래그

예 약 변 수	데 이 터 타 입	내 용
CNF_WAR	WORD	시스템의 경고(경고장)
RTC_ERR	BOOL	RTC 데이터 이상
D_BCK_ER	BOOL	데이터 백업 에러
H_BCK_ER	BOOL	핫 리스타트 수행 불가 에러
AB_SD_ER	BOOL	비정상 전원 차단(Abnormal Shutdown)
TASK_ERR	BOOL	태스크(Task) 충돌(정주기,외부 태스크)
BAT_ERR	BOOL	배터리 이상
ANNUN_WR	BOOL	외부기기의 경고장 검출

예 약 변 수	데 이 터 타 입	내 용
HSPMT1_ER	BOOL	고속 링크 파라미터 1 이상
HSPMT2_ER	BOOL	고속 링크 파라미터 2 이상
HSPMT3_ER	BOOL	고속 링크 파라미터 3 이상
HSPMT4_ER	BOOL	고속 링크 파라미터 4 이상

5) 시스템 에러 상세 플래그

예 약 변 수	데 이 터 타 입	내 용
_IO_TYER_N	UINT	모듈 타입 불일치 슬롯 넘버
_IO_TYERR	ARRAY OF BYTE	모듈 타입 불일치 위치
_IO_DEER_N	UINT	모듈 착탈 슬롯 넘버
_IO_DEERR	ARRAY OF BYTE	모듈 착탈 위치
_FUSE_ER_N	UINT	Fuse 단선 슬롯 넘버
_FUSE_ERR	ARRAY OF BYTE	Fuse 단선 슬롯 위치
_IO_RWER_N	UINT	입출력 모듈 읽기/쓰기 에러 슬롯 넘버
_IO_RWERR	ARRAY OF BYTE	입출력 모듈 읽기/쓰기 에러 슬롯 위치
_IP_IFER_N	UINT	특수/링크 모듈 인터페이스 에러 슬롯 넘버
_IP_IFERR	ARRAY OF BYTE	특수/링크 모듈 인터페이스 에러 슬롯 위치
_ANC_ERR	ARRAY OF UINT	외부기기의 중고장 검출
_ANC_WAR	ARRAY OF UINT	외부기기의 경고장 검출
_ANC_WB	ARRAY OF BIT	외부기기의 경고장 검출 비트 Map
_TC_BMAP	ARRAY OF BYTE	태스크 충돌 표시
_TC_CNT	UINT	태스크 충돌 카운터
_BAT_ER_TM	DT	배터리 전압 저하 시각
_AC_F_CNT	UINT	전원 차단 카운터
_AC_F_TM	ARRAY OF DT	순시정전 이력

6) 시스템 운전 상태 정보

예 약 변 수	데 이 터 타 입	내 용
CPU TYPE	UINT	시스템의 형태
VER NUM	UINT	PLC O/S 버전 번호
MEM TYPE	UINT	메모리 모듈의 타입
SYS STATE	WORD	PLC 모드 및 상태
GMWIN CN	BYTE	PADT 연결 상태
RST TY	BYTE	리스타트 모드 정보
INIT RUN	BIT	초기화 수행 중
SCAN MAX	UINT	최장 스캔 시간(ms)
SCAN MIN	UINT	최단 스캔 시간(ms)
SCAN CUR	UINT	현재 스캔 시간(ms)
STSK NUM	UINT	실행시간 확인을 요하는 태스크 넘버
STSK MAX	UINT	최장 태스크 실행 시간(ms)
STSK MIN	UINT	최단 태스크 실행 시간(ms)
STSK CUR	UINT	현재 태스크 실행 시간(ms)
RTC TIME	ARRAY OF BYTE	현재 시각
SYS_ERR	UINT	이상 종류

7) 통신 모듈 정보 플래그 [n 은 통신 모듈이 장착되어 있는 슬롯 번호에 해당 (n = 0 ~ 7)]

예 약 변 수	데 이 터 타 입	내 용
CnVERNO	UINT	통신 모듈의 버전 No.
CnSTNOH CnSTNOL	UINT	통신 모듈의 국번
CnTXECNT	UINT	통신 프레임 전송 에러
CnRXECNT	UINT	통신 프레임 수신 에러
CnSVCFCNT	UINT	통신 서비스 처리 에러
CnSCANMX	UINT	통신 스캔 타임 최대(1ms 단위)
CnSCANAV	UINT	통신 스캔 타임 평균(1ms 단위)
CnSCANMN	UINT	통신 스캔 타임 최소(1ms 단위)
CnLINF	UINT	통신 모듈 시스템 정보
CnCRDER	BOOL	통신 모듈의 시스템 에러(에러 = 1)
CnSVBSY	BOOL	공용 RAM 자원 부족(부족=1)
CnIFERR	BOOL	인터페이스 에러(에러 = 1)
CnINRING	BOOL	통신 참여(IN_RING = 1)

8) 리모트 I/O 제어 플래그[m 은 통신 모듈이 장착되어 있는 슬롯 번호에 해당(m = 0 ~ 7)]

예 약 변 수	데 이 터 타 입	내 용
FSMm_reset	BOOL(Write 가능)	리모트 I/O 국 리셋 제어(리셋=1)
_FSMm_io_reset	BOOL(Write 가능)	리모트 I/O 국의 출력 접점 리셋 제어(리셋=1)
FSMm_st_no	USINT(Write 가능)	해당 리모트 I/O 국의 국번호

9) 고속 링크 정보 상세 플래그 [m 은 고속 링크 파라미터의 번호(m = 1,2,3,4)에 해당]

예 약 변 수	데 이 터 타 입	내 용
HSmRLINK	BIT	고속 링크의 RUN_LINK 정보
HSmLTRBL	BIT	고속 링크의 비정상 정보(Link Trouble)
_HSmSTATE	ARRAY OF BIT	고속 링크의 파라미터에서 k 데이터 블록의 종합적 통신 상태 정보
_HSmMOD	ARRAY OF BIT	고속 링크의 파라미터에서 k 데이터 블록에 설정된 국의 모드 정보 (Run = 1, 이외 = 0)
_HSmTRX	ARRAY OF BIT	고속 링크의 파라미터에서 k 데이터 블록의 통신 상태 정보 (정상 = 1, 비 정상 = 0)
_HSmERR	ARRAY OF BIT	고속 링크의 파라미터에서 k 데이터 블록에 설정된 국의 상태 정보 (정상 = 0, 에러 = 1)

2. 예약어

예약어는 시스템에서 사용하기 위해 미리 정의한 단어들입니다. 따라서 식별자로 이 예약어를 사용할 수 없습니다.

예 약 어
ACTION ... END_ACTION
ARRAY ... OF
AT
CASE ... OF ... ELSE ... END_CASE
CONFIGURATION ... END_CONFIGURATION
데이터 타입 이름
DATE#, D#
DATE_AND_TIME#, DT#
EXIT
FOR ... TO ... BY ... DO ... END_FOR
FUNCTION ... END_FUNCTION
FUNCTION_BLOCK ... END_FUNCTION_BLOCK
평선 블록의 이름들
IF ... THEN ... ELSIF ... ELSE ... END_IF
OK
연산자 (IL 언어)
연산자 (ST 언어)
PROGRAM
PROGRAM ... END_PROGRAM
REPEAT ... UNTIL ... END_REPEAT
RESOURCE ... END_RESOURCE
RETAIN
RETURN
STEP ... END_STEP
STRUCTURE ... END_STRUCTURE
T#
TASK ... WITH
TIME_OF_DAY#, TOD#
TRANSITION ... FROM... TO ... END_TRANSITION
TYPE ... END_TYPE
VAR ... END_VAR
VAR_INPUT ... END_VAR
VAR_OUTPUT ... END_VAR
VAR_IN_OUT ... END_VAR
VAR_EXTERNAL ... END_VAR
VAR_ACCESS ... END_VAR
VAR_GLOBAL ... END_VAR
WHILE ... DO ... END_WHILE
WITH

부록 E. 용어설명

용 어	정 의	비 고
모듈 (Module)	시스템을 구성하는 일정한 기능을 가진 표준화된 요소로서 베이스에 삽입되도록 조립된 입출력 보드와 같은 장치	예) CPU 모듈, 전원 모듈, 입출력 모듈 등
유닛 (Unit)	PLC 시스템의 동작상에서 최소 단위가 되는 모듈 또는 모듈의 집합체이며, 다른 모듈 또는 모듈의 집합체와 접속되어 PLC 시스템을 구성하는 것	예) 기본 유닛, 증설 유닛
PLC 시스템 (PLC System)	PLC 와 주변장치로 이루어지는 시스템으로 사용자 프로그램에 의하여 제어가 가능하도록 구성된 것	
콜드 리스타트 (Cold Restart)	모든 데이터(입출력 이미지 영역, 내부 레지스터, 타이머, 카운터 등의 변수 및 프로그램)를 자동 또는 수동에 의하여 정해진 상태로 초기화 한 후 PLC 시스템 및 사용자 프로그램을 다시 시동하는 것	
웜 리스타트 (Warm Restart)	전원의 Off 발생을 사용자 프로그램에 통지하는 기능을 가지고, 전원 Off 가 발생한 후 사용자가 사전에 정한 데이터 및 사용자 프로그램에 따라 다시 시동하는 것	
핫 리스타트 (Hot Restart)	전원 Off 가 발생한 후 최대 허용 시간 이내에 PLC 시스템이 모든 데이터를 그 이전의 상태로 복귀시켜 다시 시동하는 것	
입출력 이미지 영역	입출력 상태를 유지하기 위하여 설치된 CPU 모듈의 내부 메모리 영역	
워치독 타이머 (Watchdog Timer)	프로그램의 미리 정해진 실행 시간을 감시하고 규정 시간 내에 처리가 완료되지 않을 때 경보를 발생하기 위한 타이머	
평선 (Function)	4 칩 연산, 비교 연산 등과 같이 연산 결과를 명령어 내부에 기억하지 않고 입력에 대한 연산 결과를 즉시 출력하는 연산 단위	
평선 블록 (Function Block)	타이머, 카운터 등과 같이 명령어 내부에 연산 결과를 기억하여 여러 스캔에 걸쳐 기억된 연산 결과를 이용하는 연산 단위	

용 어	정 의	비 고
직접 변수	이름, 타입을 별도로 선언하지 않고 사용하는 변수로 %I, %Q, %M 영역이 이 변수에 해당함	예) ·%IX0.0.2 ·%QW1.2.1 ·%MD1234 등
네임드 변수 (Named 변수)	사용자가 이름, 타입 등을 선언하고 사용하는 변수로 ‘스위치0’= %IX0.0.2, ‘결과값’=%MD1234 등과 같이 선언하면 %IX0.0.2 와 %MD1234 대신에 ‘스위치0’과 ‘결과값’의 이름으로 프로그래밍할 수 있음.	
GMWIN	프로그램 작성, 편집, 컴파일 및 디버그 기능을 수행하는 GLOFA-GM 시리즈용 로더	
싱크(Sink) 입력	입력 신호가 ON 될 때 스위치로부터 PLC 입력 단자로 전류가 유입되는 방식	
소스(Source) 입력	입력 신호가 ON 될 때 PLC 입력 단자로부터 스위치로 전류가 유입되는 방식	
싱크 출력	PLC 출력 접점이 ON 될 때 부하에서 출력 단자로 전류가 유입되는 방식	
소스 출력	PLC 출력 접점이 ON 될 때 출력 단자로부터 부하로 전류가 유입되는 방식	

부록 F. PLC 설치 환경 및 배선

(1) 설치 환경

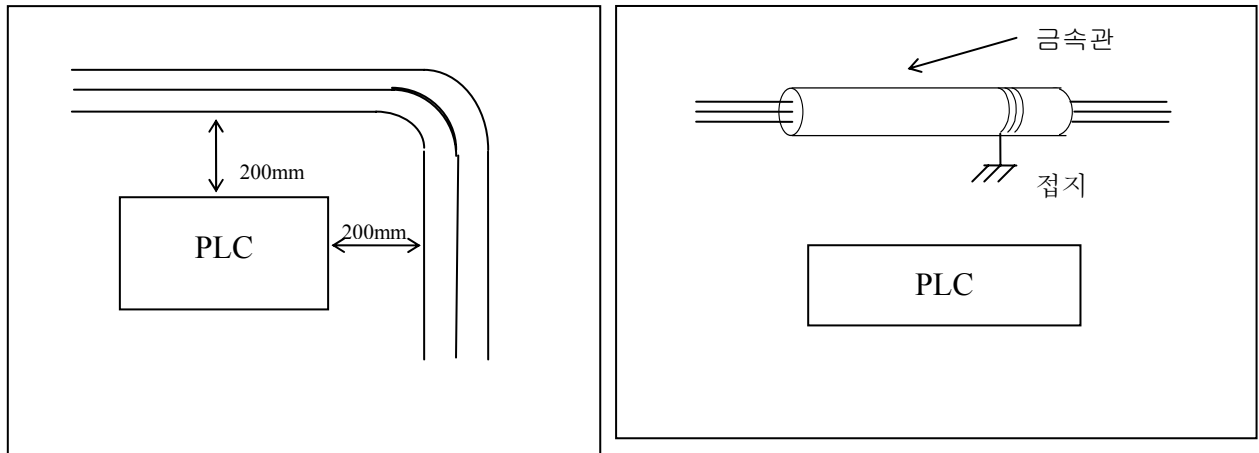
PLC 가 사용되는 주위 환경에서 고려해야 할 점은 온도, 습도, 노이즈, 진동 충격, 절연 저항, 접지 등으로, 선정된 PLC 가 설치 환경 조건에서 잘 견딜 수 있는지 검토해야 한다.

표 E-1 은 PLC 설치 환경에 대한 일반 사양이다.

표 E-1

일 반 사 양

No	항 목	규 격				관련 규격	
1	사용 온도	0 ~ 55℃					
2	보관 온도	-25 ~ 75℃					
3	사용 습도	5~95%RH, 이슬이 맺히지 않을 것					
4	보관 습도	5~95%RH, 이슬이 맺히지 않을 것					
5	내 진동	단속적인 진동이 있는 경우				IEC 1131-2	
		주파수	가속도		진폭		횟수
		10≤ f< 57 Hz	-		0.075 mm		X,Y,Z 각 방향 10 회
		57 ≤ f ≤ 150 Hz	9.8 ^{m/s²} {1 G}		-		
		연속적인 진동이 있는 경우					
		주파수	가속도		진폭		
		10≤ f< 57 Hz	-		0.035 mm		
		57 ≤ f ≤ 150 Hz	4.9 ^{m/s²} {0.5 G}		-		
6	내 충격	● 최대 충격 가속도 : 147 ^{m/s²} {15G} ● 인가시간 : 11ms ● 펄스 파형 : 정현 반파 펄스(X,Y,Z 3 방향 각 3 회)				IEC 1131-2	
7	내 노이즈	방형파 임펄스 노이즈	± 1,500 V			LG 산전 내부 시험 규격 기준	
		정전기 방전	전압 : 4Kv(접촉 방전)			IEC 1131-2, IEC 801-2	
		방사 전자계 노 이즈	27 ~ 500 MHz, 10V/m			IEC 1131-2, IEC 801-3	
		패스트 트랜지언 트/ 버스트 노이즈	구 분	전원 모듈	디지털 입출력 (24V 이상)	디지털 입출력 (24V 미만) 아날로그 입출력 통신 인터페이스	IEC 1131-2 IEC 801-4
			전 압	2Kv	1kV	0.25kV	
8	주위 환경	부식성 가스, 먼지가 없을 것					
9	냉각 방식	자연 공랭식					



(a) 이격 거리 준수

(b) 금속관에 의한 유도 노이즈 차폐

그림 E-2 유도 노이즈 방지 방법

PLC 설치시 접지 및 배선의 유의 사항 등을 지켜주면 노이즈를 어느 정도 줄일 수 있다. 이 외에 다른 전자 기기와의 영향을 줄이기 위하여 다음과 같은 대책이 필요하다.

교류 유도성 부하의 경우, 그림 E-3 와 같이 RC 또는 바리스터(Varister)를 부하 양끝에 병렬 접속한다.

직류 유도성 부하의 경우, 그림 E-3 와 같이 부하 근처에 다이오드를 역방향으로 병렬 접속한다.

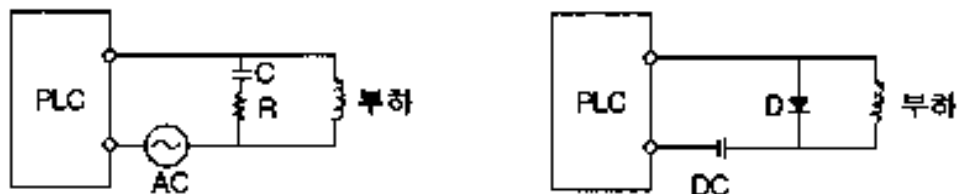


그림 E-4 유도성 부하의 노이즈 대책

일반적인 노이즈 대책

표 E-2 는 일반적인 노이즈의 종류이다.

표 E-2 일반적인 노이즈의 종류

노이즈의 종류
<ul style="list-style-type: none"> · 개폐기 노이즈(전자 개폐기, 전자 접촉기등) · 낙뢰 서지 · 입 · 출력 선이나 제어 신호선, 전송 케이블을 통하여 유도되는 노이즈 · 강전기 노이즈 · 강전자계 발생원에 의한 전자 유도 노이즈

표 E-3 일반적인 노이즈의 종류

항 목	노이즈 대책
PLC 제어반 내부	<ul style="list-style-type: none"> · 차단기, 배선용 차단기, 전자 개폐기 등 아크가 발생하는 기기와는 가능한 분리 설치 · 전원선을 꼬아서 사용 · 입력선과 출력선의 분리
PLC 와 외부 연결	<ul style="list-style-type: none"> · 차폐 변압기 · 최단 거리 배선 · 정전압 전원 · 필터
제어반의 외부 배선	<ul style="list-style-type: none"> · 실드(Shield) 케이블의 사용 · 입력선과 출력선의 분리 · 광 전송 케이블로 전송
접 지	<ul style="list-style-type: none"> · 전용 접지는 우수 · 공용 접지는 양호 · 공통 접지는 불가
기 타	<ul style="list-style-type: none"> · 서지 흡수식 전자 접촉기 · 서지 옵서버의 접속 · 릴레이 코일(DC 인 경우)에 다이오드의 접속

부록 G. 유지 보수

(1) 보전이란?

공장 자동화 시스템이 고장 없이 가동되어 생산성을 높이는 것이 PLC 사용의 최대 목표라 할 수 있다.

PLC은 반도체를 사용한 전자 회로로 반영구적이므로 릴레이 제어반 처럼 예방 보전적인 부품의 교환 처리는 필요하지 않으나 릴레이 출력 카드나 전지 등의 정기적인 교환은 필요하다. 만일 고장이 나면 모듈을 교환하면 된다.

그 밖에 시스템의 고장 요인으로는 다음 7가지가 있다.

- PLC의 하드웨어
- PLC의 소프트웨어
- PLC의 제어 및 조작반
- 기계의 검출부
- 기계의 구동부
- 기계의 본체
- 시스템 주변 기기의 환경

장치나 시스템이 가동될 때 그 기능이나 성능을 유지하기 위한 점검, 조정, 대체, 수리 등의 작업을 보전(保全)이라 하는데, 크게 예방 보전과 사후 보전의 2가지가 있다. 생산 설비, 항공기 등 경제적 손실이 크거나 중대 사고에 연결되는 것은 예방 보전이 적용되고 일반 제품은 사후 보전이 적용된다.

(2) 예방 보전

① 일상 점검

일상 점검은 PLC 본체에 관한 것과 외부에서 공급되는 전원이나 온도, 습도 등의 주위 환경에 관한 것이다. 어느 것이나 매일 운전하기 전에 점검하는 것이 바람직하며, 구체적인 점검 항목을 작성하는 것이 필요하다.

② 정기 점검

1개월, 3개월, 6개월 등의 비교적 긴 시간마다 점검하는 것으로 현상이 천천히 변화해 가기 때문에 매일 점검할 필요가 없는 것에 해당한다.

(3) 사후 보전

① 이상 발견

평소와는 다른 현상으로 동작되는 경우로서 무엇이 이상인지 원인을 명확히 찾아내는 것이 필요하다. PLC의 자기 진단에 의한 것 외에 사용자 프로그램으로 중요한 동작 과정을 진단하여 기계 장치의 이상 유무를 판단할 수 있다. 이 외의 발견 방법으로는 PLC 하드웨어 및 주변 기기의 이상 상태 체크, 기계의 움직임에 의한 이상 상태 체크, 제품의 형상이나 생산량에 의한 이상 체크 등이 있다.

② 이상 현상과 조치

이상의 발견되면 즉시 복구하여 시스템이 재가동 될 수 있도록 한다. 이때 주의할 것은 이상이 다른 곳 까지 파급되는 경우가 있으므로 다른 곳에서의 영향도 함께 진단할 필요가 있다.

(2) PLC 점검 요령

표 1 점검 항목과 내용

점검항목	점검내용	점검주기	
		일상	정기
주변환경	주변온도, 습도, 먼지, 오일미스트 등을 확인	○	
전원	Maker 지정 범위 내 인가 확인	○	
취부상태	Unit(I/O 포함) 취부 상태의 느슨해짐 정도,절단	○	
(배선)	단자, 볼트의 조임 확인	○	
	배선 Cable 의 손상, 열화확인	○	
	압착단자(cable)의 근접	○	
표지 Lamp	동작(상태)표시기의 정상동작 확인	○	
Battery	전압은 정상인가, Maker 보증 기간 내인가 (표시램프, 모니터 등에서 check)		○
Relay	동작시에 「빠리리」 흡은 없는가		○
FusE	느슨해짐, 절단은 없는가		○
Program (usersoft)	Master Priogram(보관)과 Program 내용을 비교, 조합 하여		○
제어반	상호확인		○
이물제거	냉각 Fan 및 Air-Filte 의 청소		○
예비품	먼지, 이물 등을 청소제거		○
	보관 개수 Check		○
	보관환경 Check		○
	동작 Check		

표 2

교환 부품

부품명	표준 교환 년 수	교환방법, 기타
Battery	2-3 년(단, 수명은 Maker 및 종류에 따라 다르다.)	신품과 교환
(전원회로) 평활 콘덴서	5 년	신품과 교환 Maker 와 상담 후에 결정
Relay 류		개폐전류,개폐빈도에 따라 다르기 때문에 Maker 규정에 의해 결정
Fuse	10 년	신품과 교환

PLC 가 다음의 항목에 해당할 때는 표 2 의 교환 부품의 교환 년 수 단축을 고려할 필요가 있다

- ① 온도, 습도가 높은 장소 또는 그 변화가 심한 장소에서 사용할 경우
- ② 전원(전압, 주파수, 파형 찌그러짐 등)이나 부하의 변동이 큰 경우
- ③ 진동, 충격이 심한 장소에 설치된 경우
- ④ 먼지,염분,아황산가스 및 유황수소 등의 나쁜 환경 속에서 사용할 경우
- ⑤ 사용전 보관 환경이 나쁜 경우(장기보존, 장기정지 등)

표 3

필요한 예비품

NO	품명	수량	비고
1	Batery	1-2 개	전지의 보존수명은 약 3 년이다 1-2 개는 예측할 수 없는 경우에 대비한다
2	Fuse	사용수	Fuse 는 단락이나 과전류 뿐만 아니라 전원 ON/OFF 등의 돌입전류에 의해 끊어질 수 있기 때문에 넉넉하게 준비한다

표 4 준비 권장 예비품

NO	품명	수량	비고
1	입·출력 Unir	Unit 의 각명 에 붙여 1 개	Relay 출력 Unit 는 점점마모가 있다
2	CPU	1 개	PLC 의 핵심이 되는 부품이므로 만일 고장이 났을 때에는 System 이 Down 된다
3	Memory	1 개	
4	전원 Unit	1 개	

표 5 Data 보존용 예비품

NO	품명	수량	비고
1	Print 용지	필요수 (그때마다 수배)	
2	Foppy Disk	필요수	시운전용이 Back-UP 과 User 용의 예비

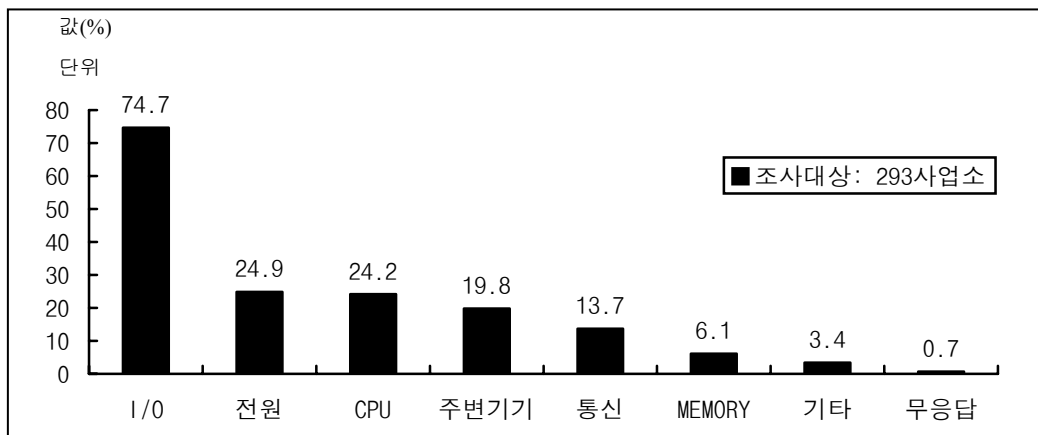


그림 1 PLC의 고장 부위(복수 응답)

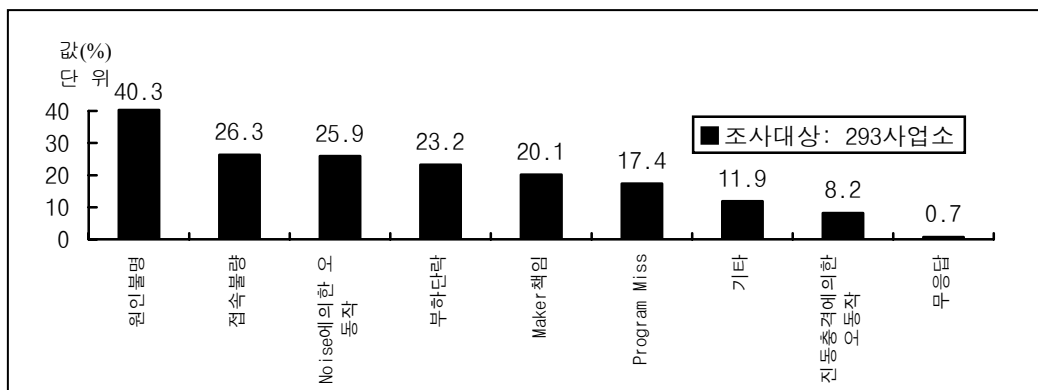


그림 2 PLC의 고장 원인(복수 응답)

LS PLC 제품 Line-Up

형명	외형	최대 I/O점수 (리포트 I/O)	처리속도/Step	프로그램 메모리	특징
XGT Series	XGK-CPUH	6,144점 (32,000점)	0.028 μ s	64K Step	XGT Series 초고속, Compact, Open Network Solution을 지향하는 신개념의 차세대 PLC • 전용 MPU 탑재로 업계 최고 CPU 처리속도 실현 (0.028 μ s/Step) • 경쟁사 대비 동급 최소 Size • 통합프로그래밍 툴 지원 (XG5000, XG-PD, XG-APM) • Open Network 기반의 System 통합 • 다양한 Network 진단 및 Monitoring 기능 • Device(R, U, Z) 추가 및 Memory 용량 증대 • 구조화 및 Task 프로그래밍 구현 (총 256개) • 최대 16축 Motion 제어
	XGK-CPUH0	3,072점 (32,000점)	0.028 μ s	32K Step	
	XGK-CPUS	3,072점 (32,000점)	0.084 μ s	32K Step	
	XGK-CPUS0	1,536점 (32,000점)	0.084 μ s	16K Step	
MASTER-K Series	K1000S	1,024	0.2 μ s	30K Step	MASTER-K Series • 국내 최대의 적용실적 • 간단하고 쉬운 전용명령 지원 (Mnemonic, Ladder) • 다양한 Open Network 지원 (Ethernet, Profibus, DeviceNet) • Windows 환경의 손쉬운 프로그래밍 툴 지원 (KGLWIN)
	K300S	1,024	0.2 μ s	15K Step	
	K200S	384	0.5 μ s	7K Step	
	K120S K80S	120 80	0.1 μ s 0.5 μ s	10K Step 7K Step	
GLOFA-GM Series	GMR	7,680	0.12 μ s	2M Byte	GLOFA-GM Series • IEC61131-3 국제표준언어 지원 (LD, IL, SFC) • 다양한 Open Network 지원 (Ethernet, Profibus, DeviceNet) • Windows 환경의 손쉬운 프로그래밍 툴 지원 (GMWIN) • 이중화 (GMR CPU) 및 Multi-CPU (GM1 CPU) 기능으로 완벽한 신뢰성 구현
	GM1	16,000	0.12 μ s	512K Byte	
	GM2	4,096	0.12 μ s	512K Byte	
	GM3	2,048	0.2 μ s	256K Byte	
	GM4 *GM4-CPUC기준	3,584	0.12 μ s	1M Byte	
	GM6	384	0.5 μ s	68K Byte	
	GM7U GM7	120 80	0.1 μ s 0.5 μ s	132K Byte 68K Byte	

LS산전

Leader in Electrics & Automation