
# 자동화 시스템 제어 실습

이 정 석
인하공업전문대학 메카트로닉스공학과

기초부터 시작하는 PLC

지멘스 S7-1500

# 강의 순서 및 내용

3.1 지멘스 PLC 프로그램 작성 개요

3.2 객체지향 프로그래밍 방식

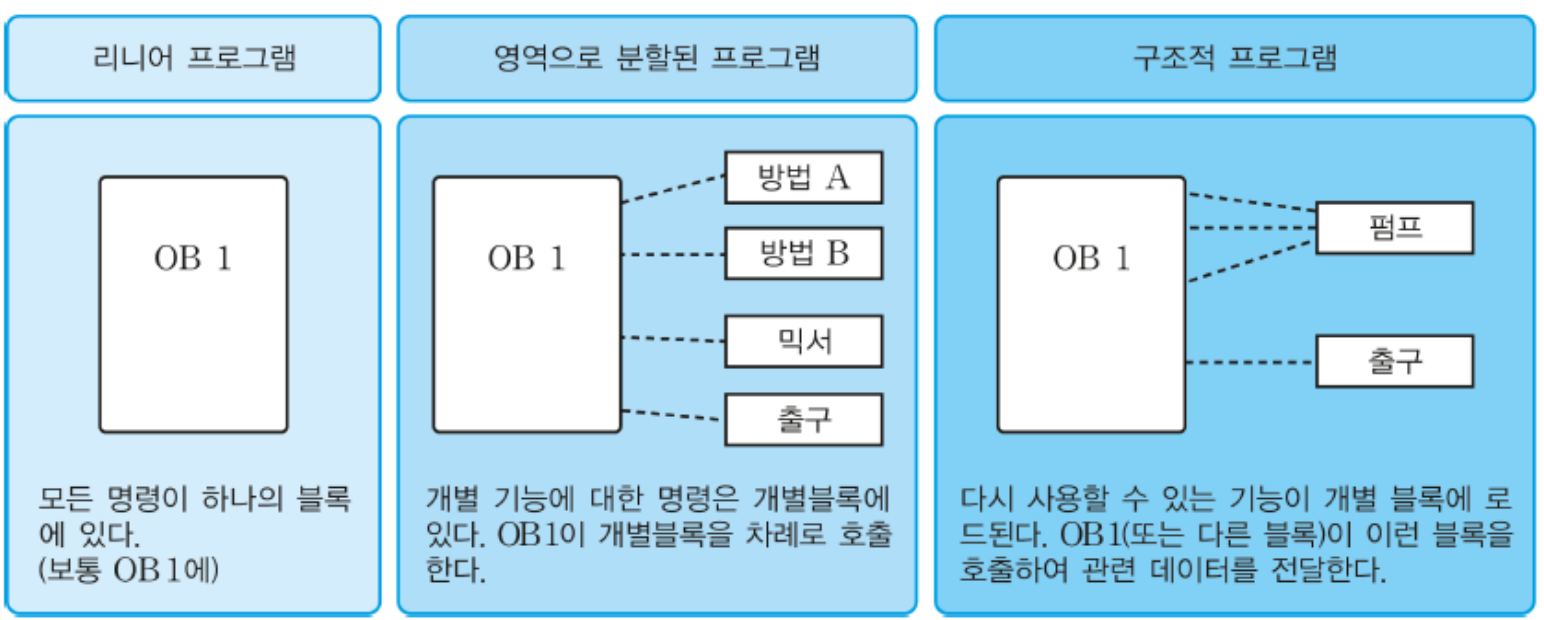
3.3 객체지향 방식을 적용한 PLC 프로그램 작성법

# 3.1 지멘스 PLC 프로그램 작성 개요

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## PLC 프로그램 작성방법 구분

- PLC의 프로그램 작성방법은 3종류로 구분
  - 주어진 프로젝트의 크기에 따라 3가지 방법 중 하나를 선택해서 사용
  - 최근에는 PLC로 제어하는 장비의 난이도와 복잡성의 증가로 구조적 프로그램 방식의 사용이 점차 확대




[그림 3-1] 지멘스 S7 PLC에서의 프로그램 작성

# 3.2 객체지향 프로그래밍 방식

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## ❖ 객체지향 프로그램 방식

➢ 객체지향 프로그램 방식은 교환형 다용도 전동공구처럼 사전에 작업의 용도에 맞는 툴을 만들어 놓고 작업의 조건에 따라 툴을 교체하거나 재조립해서 필요로 하는 작업공구를 조립하듯이 프로그램도 FB와 FC로 작성해서 OB1에서 조합해서 전체 프로그램을 만드는 방법을 의미

(a) 일체형 전동드릴 (b) 교환형 다용도 전동공구

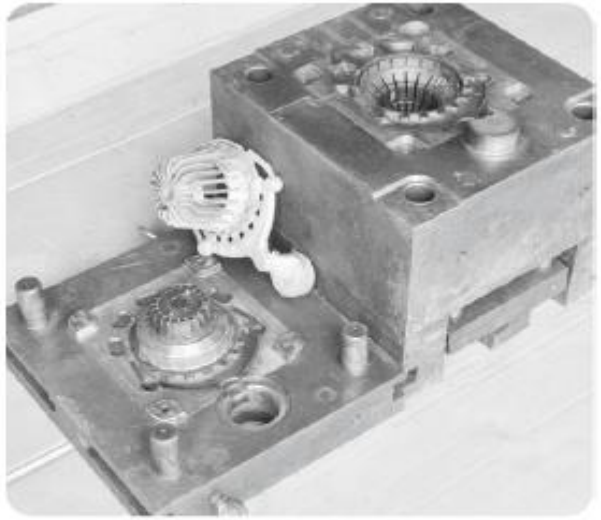
[그림 3-2] 일체형 전동드릴 및 다용도 전동공구(출처 : BOSCH 및 BLACK & DECKER 홈페이지)

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## ❖ 객체지향 프로그램 장점

➢ 객체지향 프로그램은 금형틀과 같이 필요로 하는 만큼의 제품을 만들어 낼 수 있는 것과 같이 FB와 FC를 사용해서 동작 기능별로 프로그램을 만들 경우

- 동일한 기능이 필요로 하는 곳에서 해당 FB와 FC을 따로 만들지 않고 복사해서 사용할 수 있기 때문에 복잡한 프로그램을 좀 더 쉽게 만들 수 있는 방법



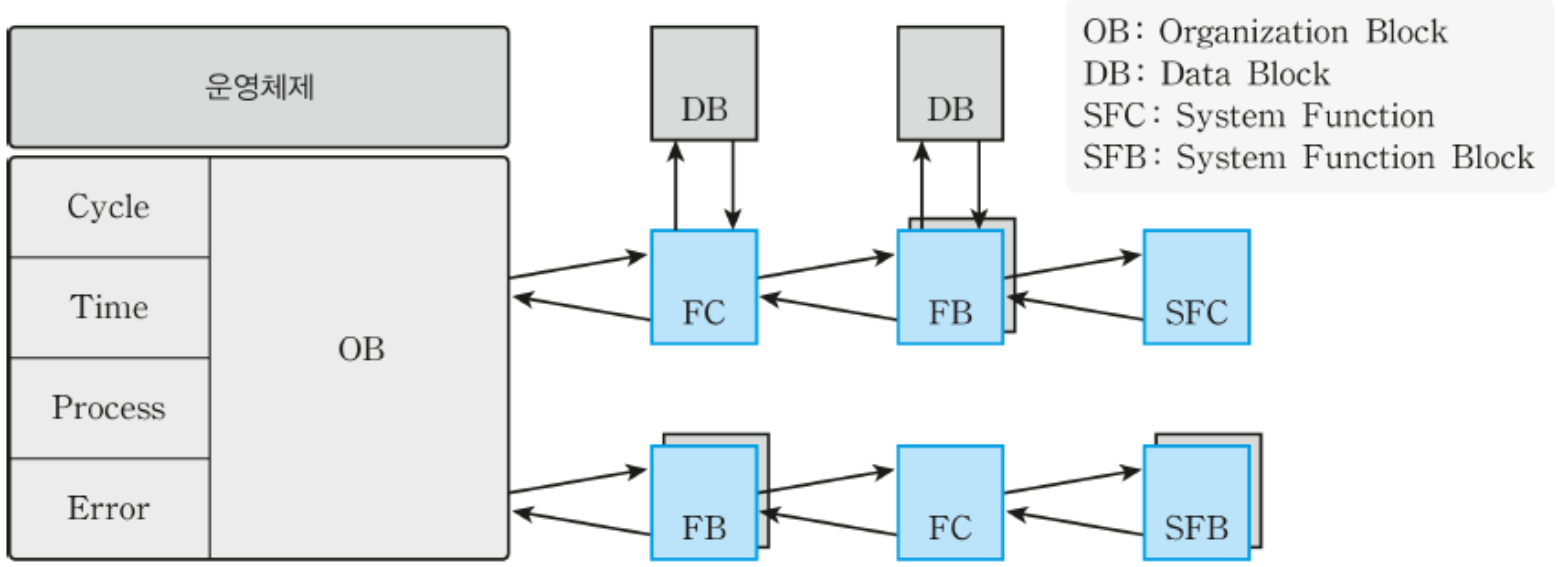
[그림 3-3] 제품을 만들기 위한 금형과 만들어진 제품

- 클래스와 객체를 사용해서 프로그램을 만듦
  - [그림 3-3]과 비교
    - 금형이 클래스에 해당
    - 금형에 의해 만들어진 제품이 객체에 해당

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## ❖ 지멘스 PLC에서의 프로그램 실행 원리

- 지멘스 PLC에서는 O/S(운영체제)에서 OB1을 매 스캔마다 호출
- OB1에서는 기능별로 작성한 FB와 FC을 호출하는 형태로 프로그램이 실행
- 지멘스 PLC에서는 프로그램을 작성할 때 사전에 면밀한 검토를 통해서 기능별로 구분한 프로그램 블록인 FB와 FC로 작성한 후에 OB1에서 조합

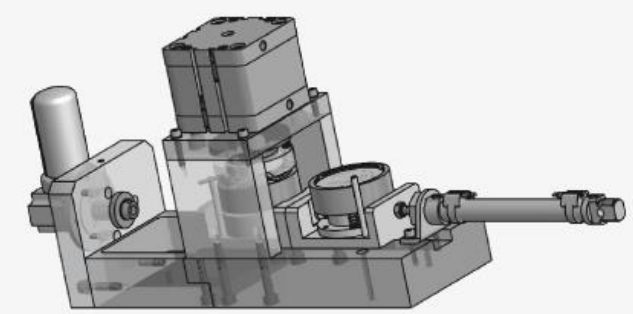



[그림 3-4] 지멘스 PLC의 프로그램 실행 방법

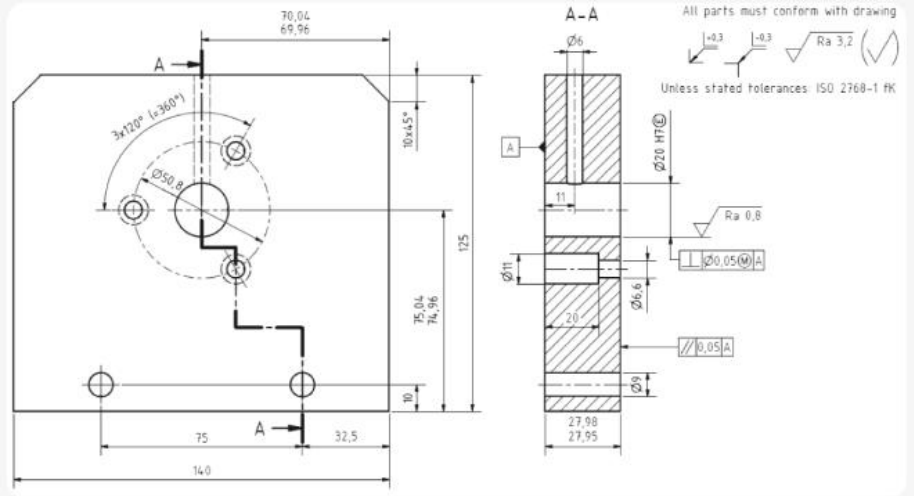
# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## ❖ 기능별로 구분한 프로그램 작성법

- ➢ 조립이 완성된 기계장치는 실제로는 수십 ~ 수백개로 구성된 부품의 조립품
  - 각각의 부품을 가공하기 위한 기계도면이 있고 도면에는 부품을 가공하기 위한 다양한 정보가 존재
- ➢ PLC프로그램도 기계장치와 같이 기능별로 구분된 다양한 FB와 FC가 존재
  - FB와 FC의 프로그램을 완성하기 위한 데이터 저장을 위한 태그와 데이터를 처리하기 위한 명령어가 존재



[그림 3-5] 조립이 완성된 기계장치

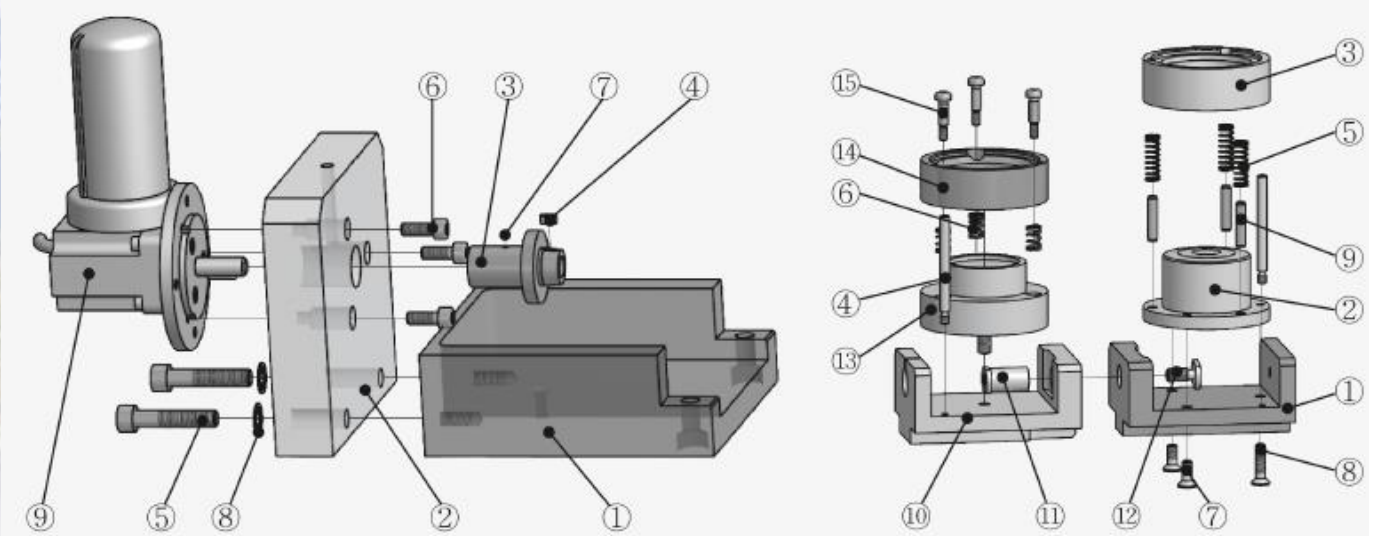



[그림 3-6] 부품 가공을 위한 설계도면

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## ❖ 기능별로 구분한 프로그램 작성법

- 기계조립에는 다양한 부품이 사용되는데 동일한 부품이 사용될 수 있음
- 프로그램에서는 동일한 부품은 메모리에 복사해서 사용할 수 있음
  - 컴퓨터에서는 동일한 파일을 다른 이름으로 복사한 파일을 손쉽게 만들 수 있음
  - 프로그램 작성에 필요한 동일한 기능인 FB와 FC가 만들어져 있다면 새롭게




[그림 3-7] 부품을 조립해서 만든 더 큰 부품

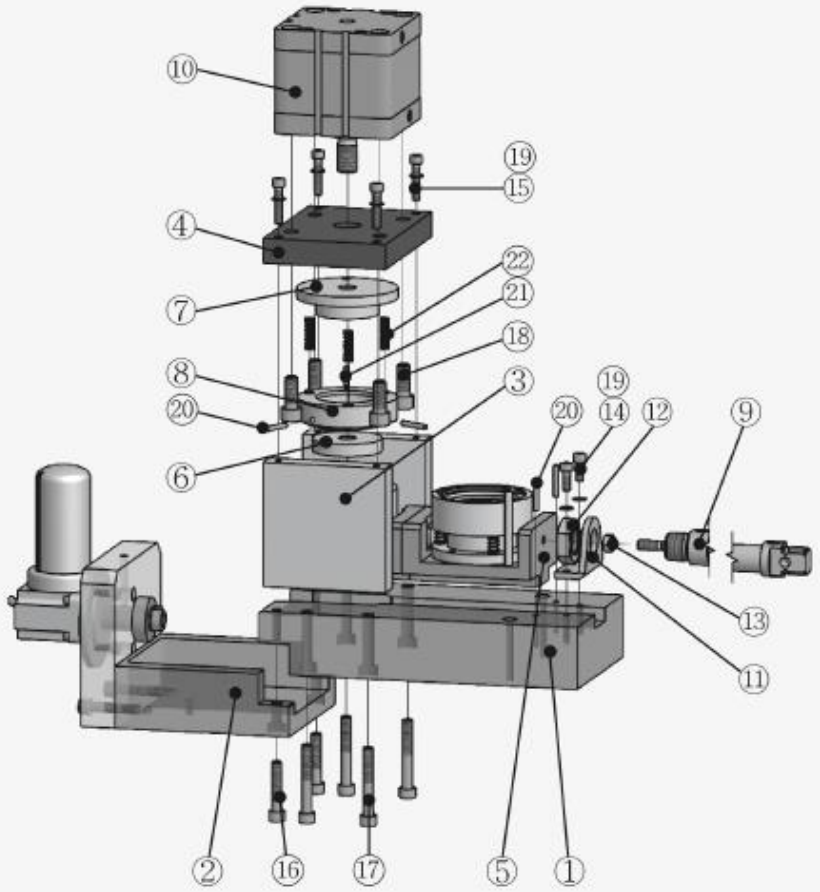
# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

❖ 기능별로 구분한 프로그램 작성법

➢ 기능별로 구분해서 만든 FB와 FC는 [그림 3-8]처럼 모든 부품이 조립되어 하나의 완제품이 만들어지듯이 OB1에서 조합되어 완성된 프로그램

➢ PLC와 기계의 차이점

- 컴퓨터가 이해할 수 있는 언어를 사용
- 작업공간으로 메모리를 사용



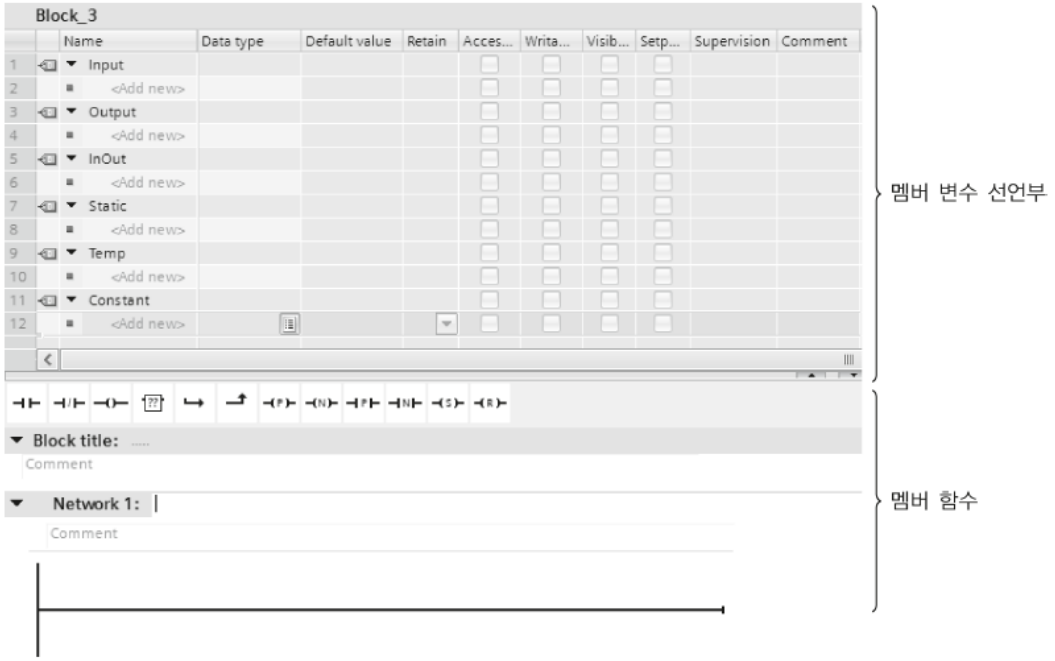

[그림 3-8] 최종 조립도면

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## ❖ 기능별 프로그램 블록 FB와 FC의 구조

➢ FB와 FC를 복사해서 사용하기 위해서는 FB와 FC가 주어진 동작을 완벽하게 구현할 수 있는 프로그램 블록이 되어야 함

- 프로그램은 데이터와 데이터를 처리하기 위한 명령어로 구성
- FB와 FC는 멤버 변수 선언부와 변수를 처리하기 위한 멤버함수로 구성



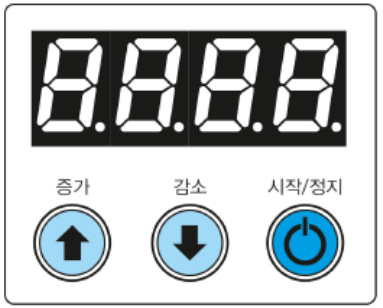

[그림 3-9] FB의 구조

# 3.3 객체지향 방식을 적용한 PLC 프로그램 작성법

# PLC 프로그램 작성 전 작업

## ❖ 시스템 동작 조건

➢ 증가 및 감소 버튼을 조작해서 0~9999까지의 숫자를 카운트하는 조작하는 패널




[그림 3-10] 조작 패널

## ❖ 동작 조건

1. 조작 패널에 전원이 인가되면 FND에는 0000이 표시
   - 단 조작 패널의 FND는 BCD 코드로 숫자를 표시
2. 시작/정지 버튼을 누를 때마다 FND에는 1과 0이 표시
   - 1이 표시될 때에는 증가, 감소 버튼에 의해 FND에 표시되는 숫자의 증가나 감소가 가능
   - 0이 표시될 때에는 증가, 감소 버튼의 조작이 불가능
3. 시작/정지 버튼에 의해 증가, 감소 버튼의 조작이 가능한 상태에서 증가나 감소 버튼을 누를 때마다 숫자는 1씩 증가 또는 감소
   - 1~9999까지의 범위 내에서 숫자 조작이 가능

# PLC 프로그램 작성 전 작업

[표 3-1] 입출력 번지 할당표

| 입력 (INPUT) | | | 출력 (OUTPUT) | | |
|---|---|---|---|---|---|
| 입력번지 | 명칭 | 비고 | 출력번지 | 명칭 | 비고 |
| I0.0 | UP_PB | 증가 | QB0 | FND | 1, 10 |
| I0.1 | DN_PB | 감소 | QB1 | FND | 100, 1000 |
| I0.2 | POW_PB | 시작/정지 | | | |

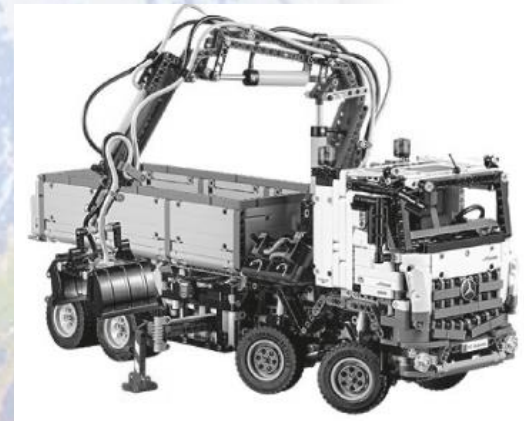
❖ 프로그램 작성을 위한 동작 조건 분석

➢ PLC 프로그램을 작성하기 위해서는 주어진 동작 조건을 분석해서 몇 개의 프로그램 블록으로 구분할 것인가를 우선 고민

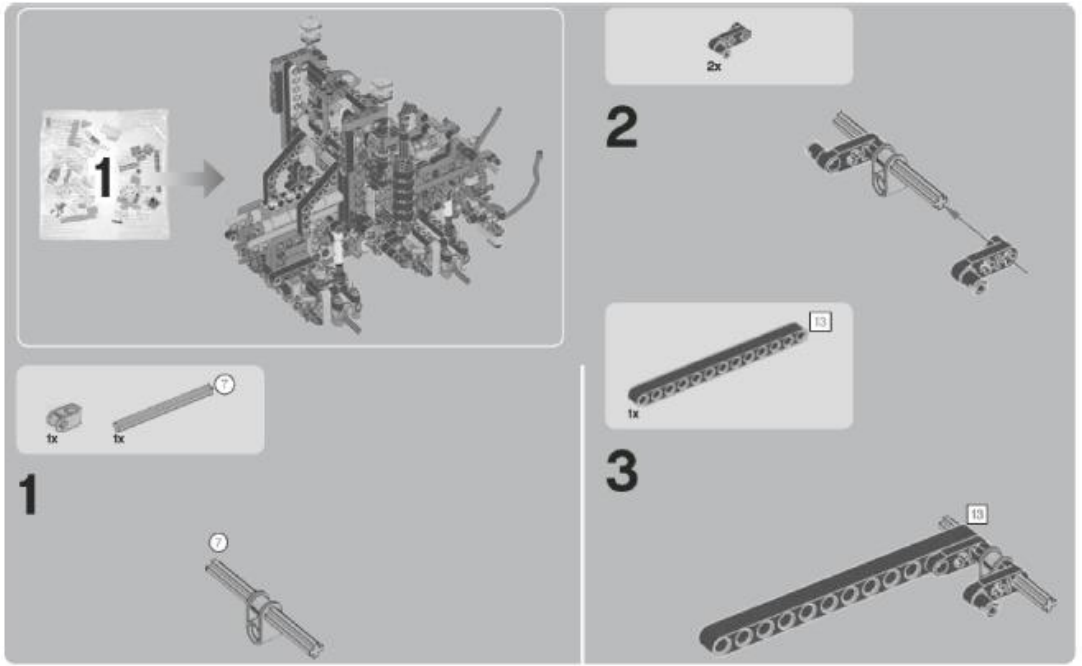
# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## PLC 프로그램 작성 전 작업

- 복잡한 구조 기계장치도 단순한 부품의 조합으로 이루어짐
  - [그림 3-11]의 레고로 만든 벤츠 트럭도
  - [그림 3-12]의 조립도와 같이 단순한 부품의 조합으로 기능 블록을 만듦
  - 여러 개의 기능블록의 조합을 통해 벤츠 트럭이 만들어짐



[그림 3-11] 레고 테크닉 벤츠 트럭



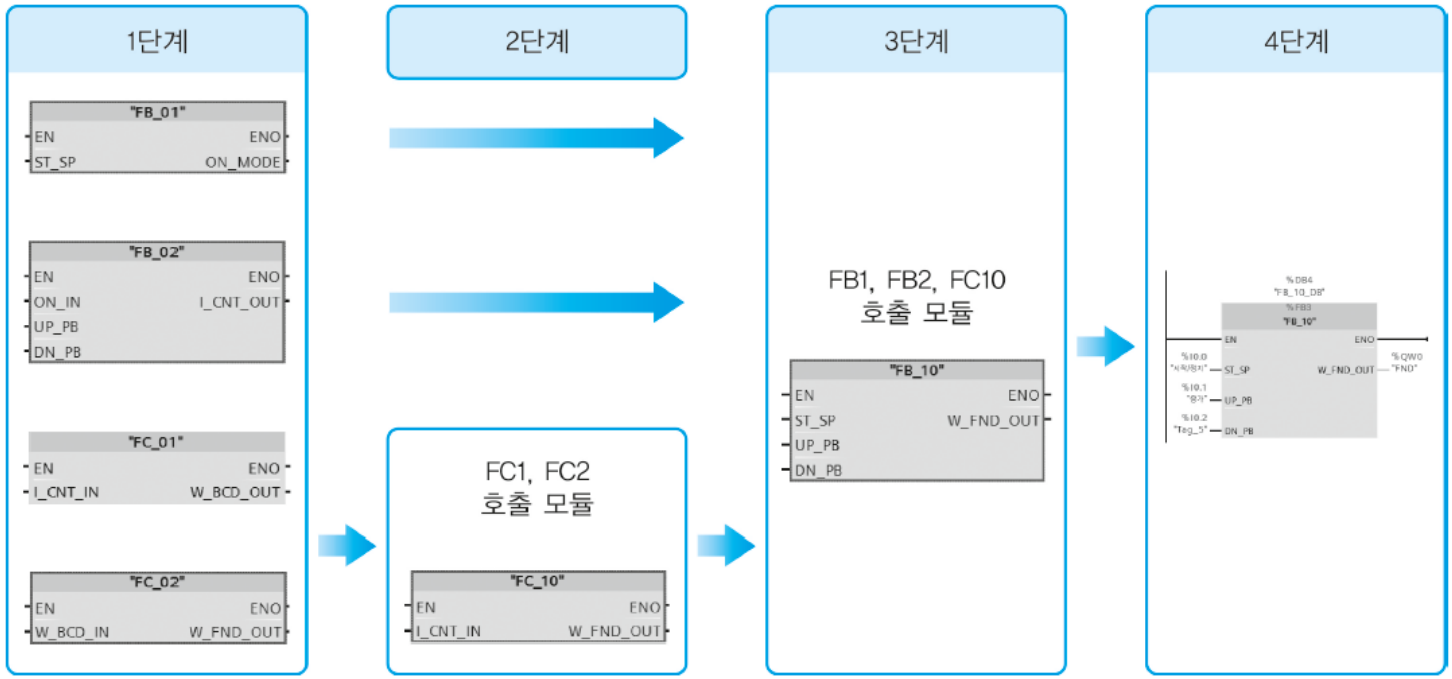

[그림 3-12] 레고 조립 설명서

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## PLC 프로그램 작성 전 작업

➢ 동작 조건을 분석하여 프로그램의 구조를 설계

- 주어진 동작 조건을 완성하기 위한 프로그램의 구조를 생각
- 1단계에서 동작조건을 분석해서 작은 기능을 가진 FB와 FC을 만듦
- 여러 단계를 거쳐 가면서 주어진 동작을 구현할 수 있는 최종 프로그램 블록을 만들고 OB1에서 호출해서 실행




[그림 3-13] 동작 조건을 분석해서 설계한 프로그램의 구성

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## PLC 프로그램 작성 전 작업

### ➢ FB와 FC의 차이점

- 태그를 선언하는 인터페이스의 구조에서 차이
- FB에는 Static의 변수를 선언할 수 있는데 FC에는 Static의 변수를 선언할 수 없음
- FC는 여러 스캔에 걸쳐서 데이터를 보관할 수 없음

| | | Name | Data type |
|---|---|---|---|
| 1 | | Input | |
| 2 | | <Add new> | |
| 3 | | Output | |
| 4 | | <Add new> | |
| 5 | | InOut | |
| 6 | | <Add new> | |
| 7 | | Static | |
| 8 | | <Add new> | |
| 9 | | Temp | |
| 10 | | <Add new> | |
| 11 | | Constant | |
| 12 | | <Add new> | |

(a) FB의 인터페이스 구조

| | | Name | Data type |
|---|---|---|---|
| 1 | | Input | |
| 2 | | <Add new> | |
| 3 | | Output | |
| 4 | | <Add new> | |
| 5 | | InOut | |
| 6 | | <Add new> | |
| 7 | | Temp | |
| 8 | | <Add new> | |
| 9 | | Constant | |
| 10 | | <Add new> | |
| 11 | | Return | |
| 12 | | Block_4 | Void |

(b) FC의 인터페이스 구조

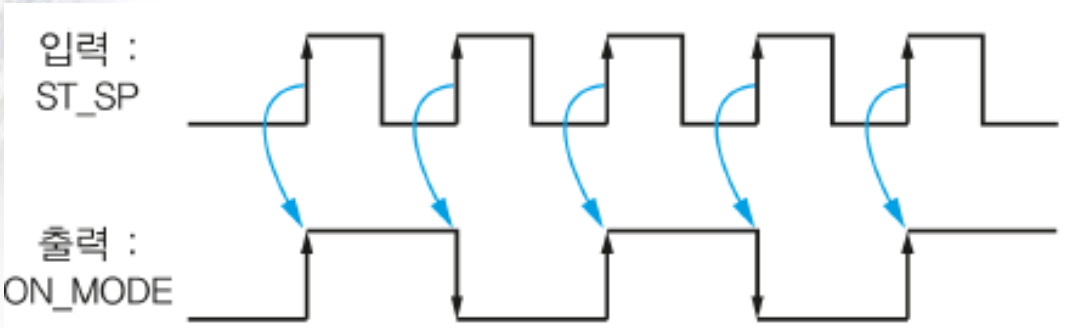
[그림 3-14] FB와 FC의 인터페이스 구조의 차이점

# PLC 프로그램 작성 전 작업

## ❖ 단계별 프로그램 모듈화 작업

### 1단계 동작 조건을 구현하기 위한 프로그램 블록 구분

- 주어진 동작 조건을 분석한 후 어떤 기능별로 구분해서 프로그램을 작성할 것인가를 결정하는 단계
- FB1의 동작
  - FB1은 START/STOP의 푸시 버튼이 ON될 때 마다 출력(ON_MODE)이 ON과 OFF 동작을 반복
  - 입력 신호는 상승펄스를 사용





[그림 3-15] FB1의 동작

# PLC 프로그램 작성 전 작업

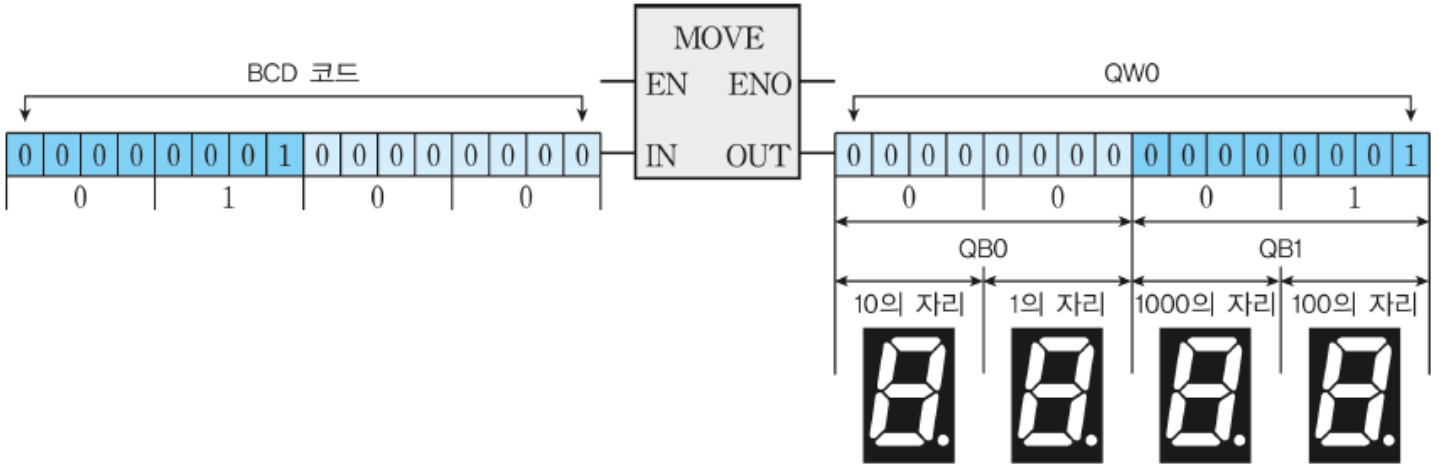
- FB_02의 동작


  - FB_02는 FB_01의 출력 'ON_MODE'가 ON 상태일 때
  - 증가나 감소 버튼의 조작에 의해 푸시 버튼의 눌린 횟수를 증가 또는 감소시키는 카운터 동작을 통해 푸시 버튼이 눌린 횟수를 출력하는 기능

- FC_01의 동작
  - FB_02의 출력 CNT_OUT(Int형)의 값을 입력으로 받아서 BCD 코드로 변환
  - 16비트 크기의 워드 값으로 BCD 코드를 출력하는 기능을 가짐
    - FC로 프로그램을 작성
    - 입력값으로부터 출력 값을 구하는 것만이 중요한 동작이기 때문

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

- FC_02의 동작
  - QB0에 1과 10의 자리에 해당되는 FND가 연결
  - QB1에 100과 1000의 자리를 표시하는 FND가 연결되어 있는 상태
    - 16비트 크기의 BCD 코드 '0100'을 출력하면 FND에는 '0001'로 표시
  - 워드 크기의 출력 데이터에서 상 하위 바이트의 자리 교환이 필요
    - FND에 표시할 PLC의 출력 데이터 형식이 서로 맞지 않기 때문
  - FC_02는 FC_01의 BCD 변환값을 입력 받은 후 값의 상 하위 바이트의 자리를 교환하는 기능을 가짐




[그림 3-16] FC2의 기능(상위 및 하위 바이트 데이터 교환의 필요성)

# PLC 프로그램 작성 전 작업

**2단계 및 3단계** 블록의 조합을 통한 또 다른 블록 생성

- 1단계에서 만들어진 FB와 FC를 조합해서 또 다른 FB와 FC를 만드는 단계
- 지멘스 PLC 프로그램에서 FB와 FC는 클래스에 해당
- 클래스의 상속
  - 클래스는 다른 클래스의 멤버 변수와 멤버 함수를 포함할 수 있는 기능
  - 객체지향 프로그래밍의 특징

- FC_10의 기능
  - FC_01은 정수를 BCD코드로 변환하는 기능
  - FC_02는 BCD코드를 FND에 표시하기 위해 상위 및 하위 바이트의 값을 변경하는 기능
  - FC_01과 FC_02 를 조합해서 정수값을 입력 받아서 FND에 숫자를 표시하는 기능을 가진 FC_10을 만듦

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## PLC 프로그램 작성 전 작업


[그림 3-17] 객체지향의 상속 개념을 이용한 FC10 만들기

# PLC 프로그램 작성 전 작업

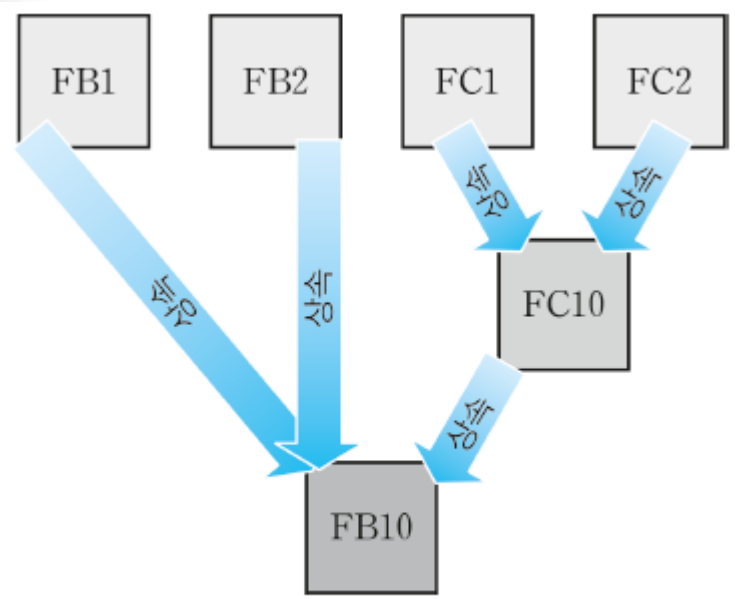
- FB_10 만들기
  - FB_01, FB_02, FC_10의 조합으로 만들어지는 OB1에서 호출할 최종 프로그램 블록


[그림 3-18] 객체지향의 상속 개념을 이용한 FB10 만들기

- 기능별로 클래스를 구분해서 FB와 FC 형태로 프로그램을 작성해야 하는 이유
  - 기능별로 작성한 FB와 FC를 상속해서 또 다른 기능을 가진 FB와 FC를 만들 수 있음(부품화)

# PLC 프로그램 작성 전 작업

- 상속원의 클래스를 '슈퍼 클래스'
- 상속하여 작성된 클래스를 '서브 클래스'
  - 서브 클래스는 얼마든지 만들 수 있지만 상속원 클래스는 하나뿐
- 상속된 클래스의 구조는 [그림 3-19]처럼 전체적으로 트리 구조



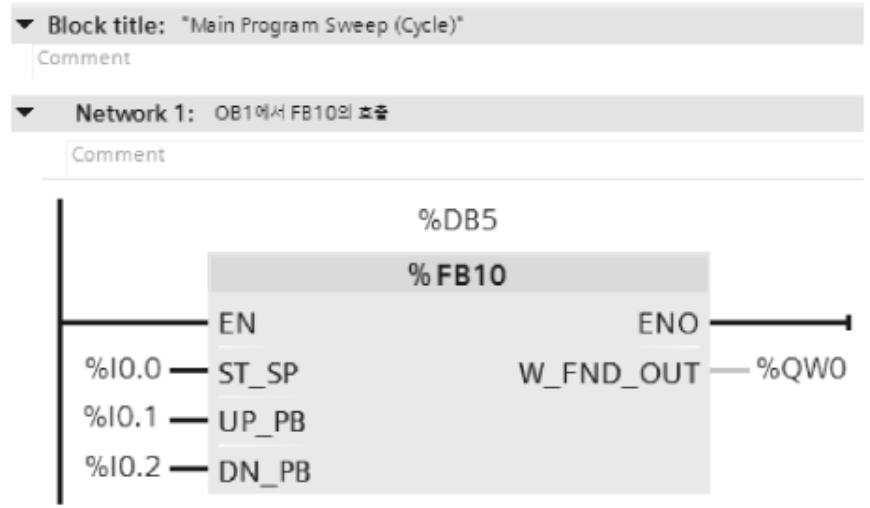

[그림 3-19] 클래스 계층과 상속 개념

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## PLC 프로그램 작성 전 작업

### 4단계 클래스의 객체화

- FB_01, FB_02, FC_01, FC_02를 상속받은 클래스 FB_10이 OB1에서 호출될 때
  - FB_10은 지멘스 PLC의 워크 메모리에서 실행
    - 해당 프로그램 블록은 프로그램 실행에 필요한 CPU의 메모리를 할당
    - 할당받은 메모리를 인스턴스 DB에 보관
    - OB1에서 FB_10의 실행에 필요로 하는 PLC의 입출력 번지를 할당
  - P를 객체라 함




[그림 3-20] OB1에서 FB10 호출

# PLC 프로그램 작성 절차(FB_01의 프로그램 작성)

## ❖ FB_01

- 1번 푸시 버튼 입력을 받으면 출력 비트로 토글 동작을 하는 기능을 가진 블록

### 1) 프로그램 작성에 필요한 입출력 태그와 데이터 타입을 결정

[표 3-2] FB_01의 구성

| FB_01 외형 | 명칭 | 구분 | 타입 | 기능 |
|---|---|---|---|---|
| "FB_01"<br>EN — ENO<br>ST_SP — ON_MODE | ST_SP | IN | BOOL | 시작/정지 푸시 버튼 입력 |
| | ON_MODE | OUT | BOOL | 토글 출력 |

# PLC 프로그램 작성 절차(FB_01의 프로그램 작성)

2) FB_01을 만들고 인터페이스에 입력과 출력 그리고 프로그램 내부에서 사용할 Static 태그를 선언

[표 3-3] FB_01의 프로그램 작성에 필요한 멤버 변수 선언

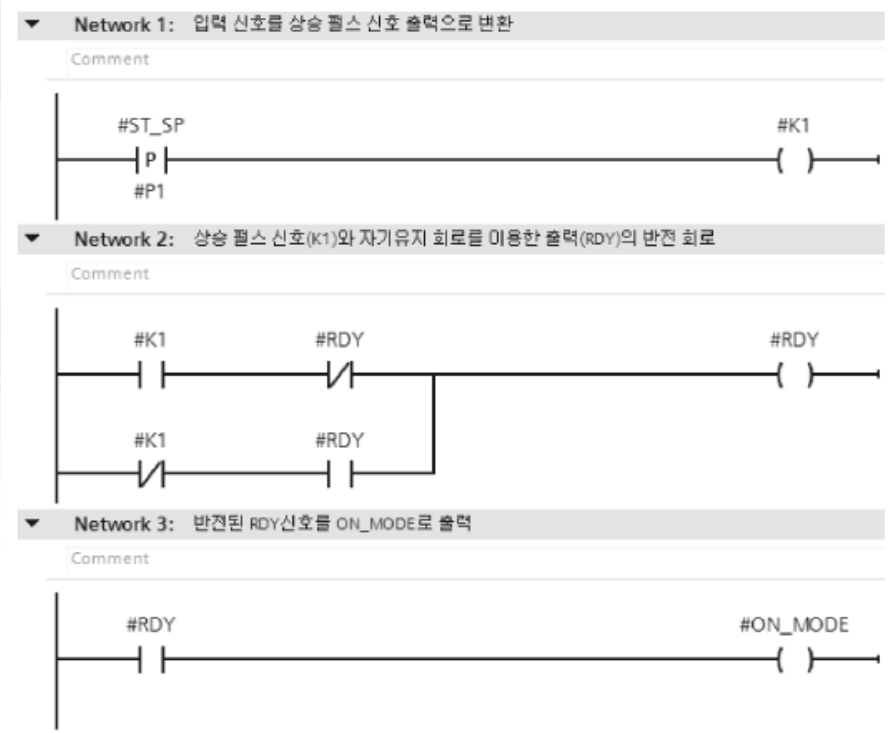
| 인터페이스 | | | | | |
|---|---|---|---|---|---|
| IN | | OUT | | STAT | |
| 변수명 | 데이터 타입 | 변수명 | 데이터 타입 | 변수명 | 데이터 타입 |
| ST_SP | BOOL | ON_MODE | BOOL | K1 | BOOL |
| | | | | P1 | BOOL |
| | | | | RDY | BOOL |

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## PLC 프로그램 작성 절차(FB_01의 프로그램 작성)

3) 입력 버튼에 의해 출력의 ON/OFF동작을 제어하기 위한 프로그램을 작성

- 해당 프로그램을 복사해서 사용하기 위해서는 반드시 내부 태그(태그 명칭 앞에 #)만을 사용해서 작성




[그림 3-21] FB_01의 프로그램

# PLC 프로그램 작성 절차(FB_02의 프로그램 작성)

## ❖ FB_02

1) FB_02는 FB_01의 출력이 ON되어 있을 때 증가와 감소 버튼에 의해 설정 값을 1씩 증가 또는 감소시키는 기능 동작을 하는 프로그램 블록

➢ 프로그램 작성에 필요한 입출력 태그와 태그 타입을 정의

[표 3-4] FB_02의 구성

| FB_02 외형 | 명칭 | 구분 | 타입 | 기능 |
|---|---|---|---|---|
| "FB_02"<br>EN — ENO<br>ON_IN — I_CNT_OUT<br>UP_PB<br>DN_PB | ON_IN | IN | BOOL | 카운터 동작 ON/OFF |
| | UP_PB | IN | BOOL | 카운터 값 증가 |
| | DN_PB | IN | BOOL | 카운터 값 감소 |
| | I_CNT_OUT | OUT | INT | 카운터 값 출력 |

# PLC 프로그램 작성 절차(FB_02의 프로그램 작성)

2) FB_01을 만들고 인터페이스에 입력과 출력 그리고 프로그램 내부에서 사용할 Static 태그를 선언

[표 3-5] FB_02의 프로그램 작성에 필요한 멤버 변수 선언

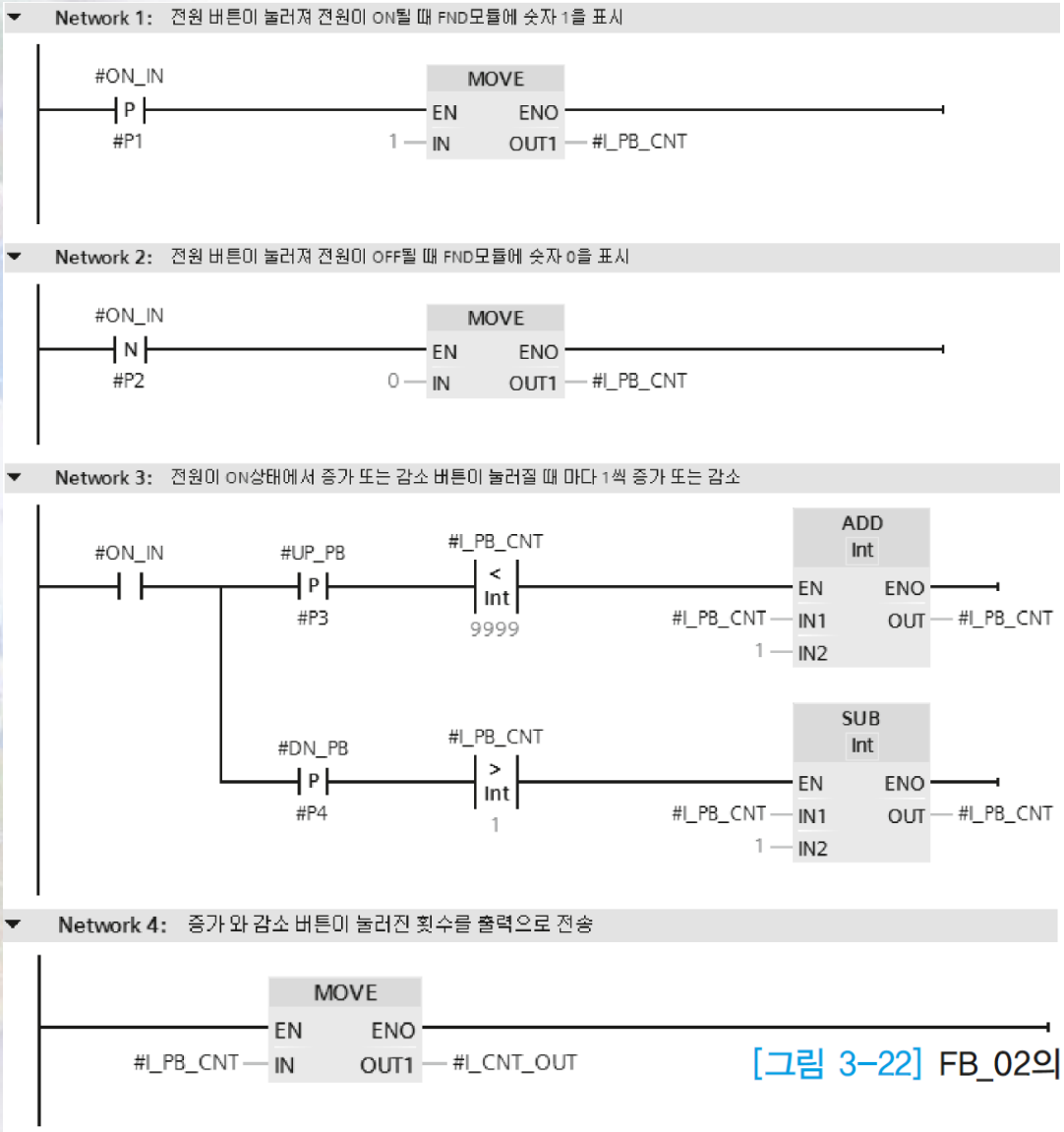
| 인터페이스 | | | | | |
|---|---|---|---|---|---|
| Input | | Output | | Static | |
| 변수명 | 데이터 타입 | 변수명 | 데이터 타입 | 변수명 | 데이터 타입 |
| ON_IN | BOOL | I_CNT_OUT | INT | I_PB_CNT | INT |
| UP_PB | BOOL | | | P1 | BOOL |
| DN_PB | BOOL | | | P2 | BOOL |
| | | | | P3 | BOOL |
| | | | | P4 | BOOL |

3) ON_IN입력이 ON될 때 설정값을 1로 만듦

- ON_IN입력이 ON상태일 때 UP_PB와 DN_PB가 눌러질 때 마다 설정값을 1씩 증가 또는 감소하는 프로그램을 작성

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## PLC 프로그램 작성 절차(FB_02의 프로그램 작성)




[그림 3-22] FB_02의 증가 및 감소 동작 프로그램

# PLC 프로그램 작성 절차(FC_01의 프로그램 작성)

## ❖ FC_01

1) FB_01은 정수 값을 입력 받아서 BCD코드로 변환하는 기능을 구현

- 입력과 출력 태그를 결정하고 데이터 타입을 정함
- 입력은 정수이고 출력은 BCD코드이기 때문에 워드 타입이 됨

[표 3-6] FC_01의 구성

| FB_01 외형 | 명칭 | 구분 | 타입 | 기능 |
|---|---|---|---|---|
| %FC1<br>"FC_01"<br>EN — ENO —<br>I_CNT_IN — W_BCD_OUT — | CNT_IN | IN | INT | 스위치가 눌린 횟수 |
| | W_BCD_OUT | OUT | WORD | 눌린 횟수를 BCD 코드로 변환 |

# PLC 프로그램 작성 절차(FC_01의 프로그램 작성)

2) FC_01을 만들고 인터페이스에 입력과 출력 그리고 프로그램 내부에서 사용할 Static 태그를 선언

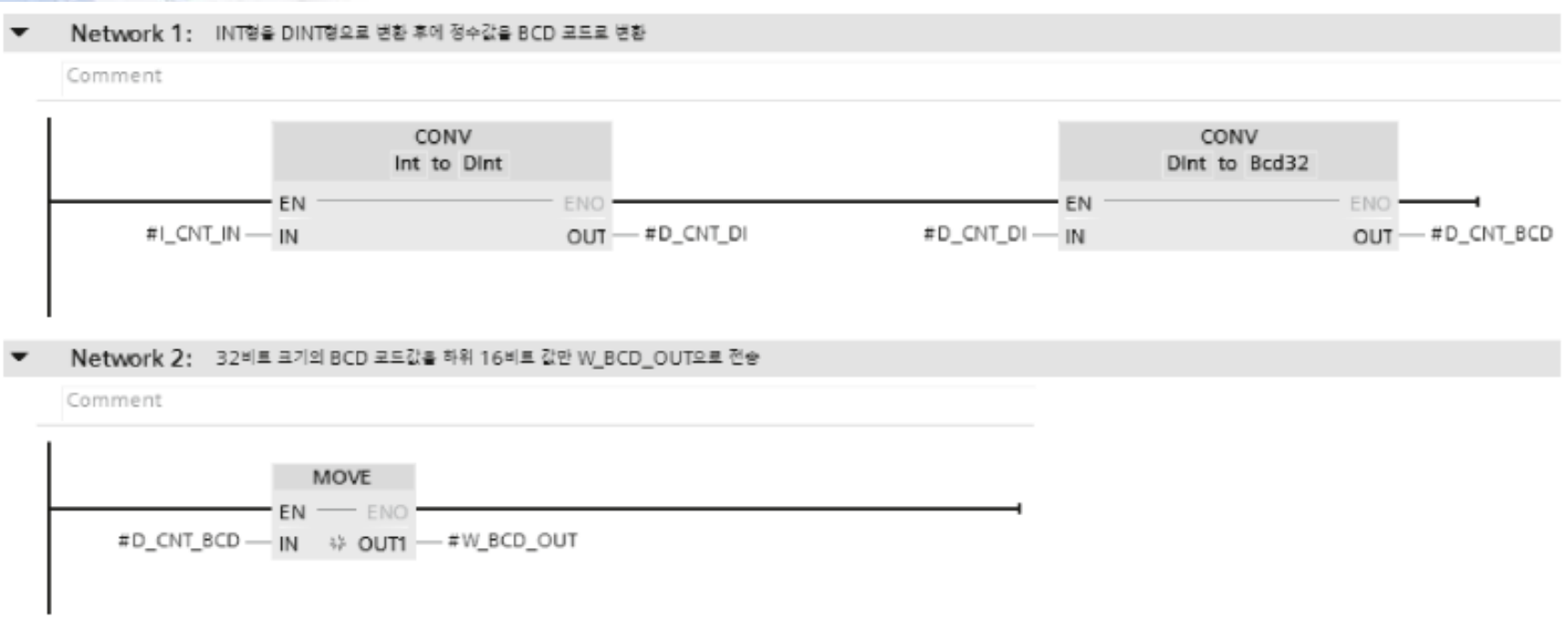
[표 3-7] FC_01의 프로그램 작성에 필요한 멤버 변수 선언

| 인터페이스 | | | | | |
|---|---|---|---|---|---|
| IN | | OUT | | TEMP | |
| 변수명 | 데이터 타입 | 변수명 | 데이터 타입 | 변수명 | 데이터 타입 |
| I_CNT_IN | INT | W_BCD_OUT | WORD | D_CNT_DI | DINT |
| | | | | D_CNT_BCD | DWORD |

# PLC 프로그램 작성 절차(FC_01의 프로그램 작성)

3) 입력 정수값을 BCD코드로 변환하기 위한 프로그램을 작성

- 9999까지의 BCD코드로 변환하기 위해 16비트 정수값을 32비트 정수값으로 변환한 후에 BCD코드로 변환하는 과정
  - 16비트 BCD코드는 +/- 999까지의 값만을 변환하기 때문




[그림 3-23] FC_01의 프로그램

# PLC 프로그램 작성 절차(FC_02의 프로그램 작성)

## ❖ FC_02

1) FC_02는 빅 엔디안 방식의 BCD코드를 리틀 엔디안 방식으로 변경하는 기능을 가진 프로그램 블록

➢ 프로그램 실행에 필요한 입력과 출력을 결정

[표 3-8] FC_02의 구성

| FC_02 외형 | 명칭 | 구분 | 타입 | 기능 |
|---|---|---|---|---|
| %FC2<br>"FC_02"<br>EN — ENO —<br>W_BCD_IN — W_FND_OUT — | W_BCD_IN | IN | WORD | 빅 엔디안 방식 BCD |
| | W_FND_OUT | OUT | WORD | 리틀 엔디안 방식 BCD |

# PLC 프로그램 작성 절차(FC_02의 프로그램 작성)

2) FC_02에 사용할 태그를 인터페이스에 선언

- 단순한 데이터 형식의 변환이기 때문에 내부 태그를 사용하지 않고 입력과 출력 태그만을 사용
- 상위와 하위 바이트의 변환은 STL명령어 CAW을 사용

[표 3-9] FC_02의 프로그램 작성에 필요한 멤버 변수 선언(STL CAW 명령 사용)

| 인터페이스 | | | | | |
|---|---|---|---|---|---|
| IN | | OUT | | TEMP | |
| 변수명 | 데이터 타입 | 변수명 | 데이터 타입 | 변수명 | 데이터 타입 |
| W_BCD_IN | WORD | W_FND_OUT | WORD | | |

# PLC 프로그램 작성 절차(FC_02의 프로그램 작성)

- CAW명령은 32비트 CPU의 ACCU레지스터에 저장된 데이터의 하위 16비트에서 상위와 하위바이트의 자리를 변경하는 하는 명령어
  - 실행 전에 ACCU에 저장된 하위 16비트의 데이터가 CAW명령어 실행 후 ACCU에 위치가 변경되어 저장

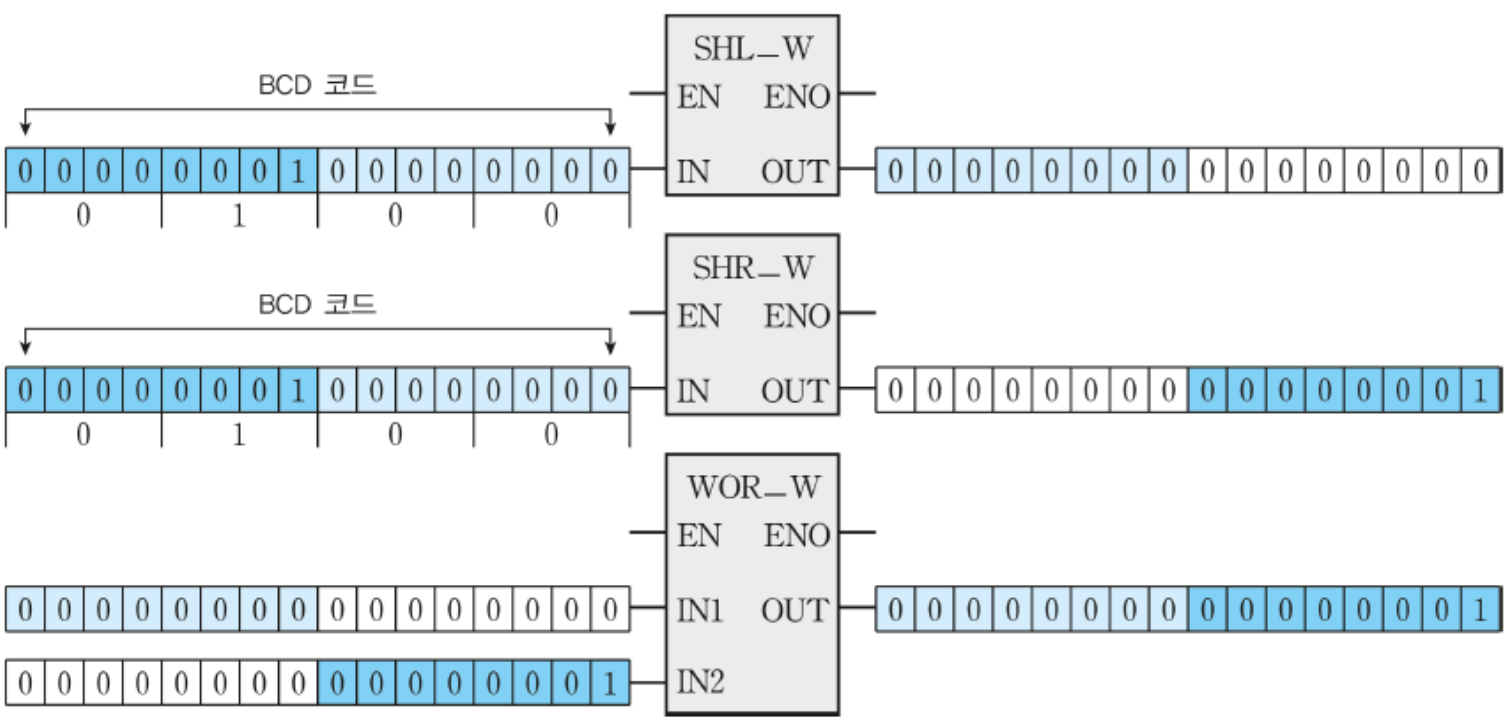
[표 3-10] CAW 명령에 의한 ACCU 레지스터의 상태 변화

| | ACCU1_HH | ACCU1_HL | ACCU1_LH | ACCU1_LL |
|---|---|---|---|---|
| 실행 전 | W4 | W3 | W2 | W1 |
| 실행 후 | W4 | W3 | W1 | W2 |

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## PLC 프로그램 작성 절차(FC_02의 프로그램 작성)

- CAW명령어를 사용하지 않고 LAD로 프로그램을 작성
  - 여러 번의 데이터 처리를 거쳐야 함
  - S1500에서는 FB, FC내부에서 네트워크 단위로 다양한 명령어 언어를 사용 가능




[그림 3-26] LAD 명령어를 사용한 상하위 바이트 자리 교환 방법

# PLC 프로그램 작성 절차(FC_02의 프로그램 작성)

- CAW명령어 대신에 LAD명령어를 사용해서 빅 엔디안 방식의 데이터를 리틀 엔디안 방식으로 변경하는 프로그램 블록
  - FC_03은 FC_02와 동일한 기능을 가지고 있음

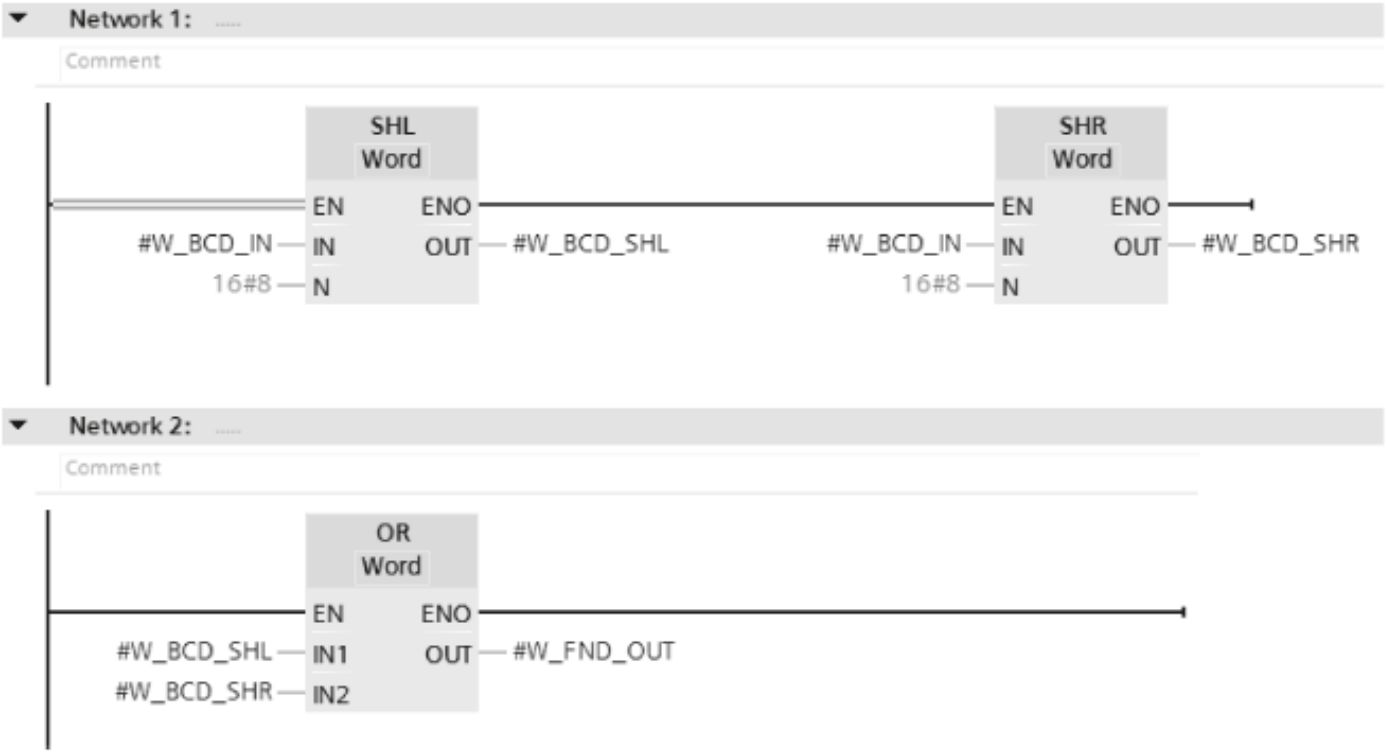
[표 3-11] FC_03의 프로그램 작성에 필요한 멤버 변수 선언(LAD 명령 사용)

| 인터페이스 | | | | | |
|---|---|---|---|---|---|
| IN | | OUT | | TEMP | |
| 변수명 | 데이터 타입 | 변수명 | 데이터 타입 | 변수명 | 데이터 타입 |
| W_BCD_IN | WORD | W_FND_OUT | WORD | W_BCD_SHL | WORD |
| | | | | W_BCD_SHR | WORD |

# 자동화시스템제어 실습(4주차) PLC 프로그램 작성

## PLC 프로그램 작성 절차(FC_02의 프로그램 작성)

- [그림 3-26]과 같은 순서로 명령어를 사용해서 CAW기능을 구현




[그림 3-27] LAD 명령을 사용해서 작성한 상하위 바이트 자리 변경 프로그램


Thank You