Ezi-MotionGate

EtherCAT

PLC 예제 (OMRON NJ-Series)

ver.1.01.03.(rev.12) 2013-05-03

IO-Map 1.5.3

목 차

FB 예 제

ST 명 령

※ 사용하시기 전에 ※

- 파스텍 Ezi-MOTIONGATE를 구입해 주셔서 대단히 감사합니다.
- Ezi-MOTIONGATE는 32bit 고성능 ARM 프로세서를 탑재한 Fieldbus to FASTECH protocol Gateway Unit입니다.
- 이 사용자 설명서에는Ezi-MOTIONGATE의 취급 방법, 안전상의 주의 사항, 이상진단과 처치방법 및 사양 등이 기재 되어있습니다.
- 사용자 설명서를 잘 이해하신 후에 Ezi-MOTIONGATE를 안전하게 사용하여 주십시오.
- 사용자 설명서를 다 읽으신 후에는 본 제품을 사용하는 사람이 언제든지 볼 수 있도록 잘 보관해 주십시오.

1. 안전상의 주의사항

◆ 일반 주의사항

- 사용자 설명서는 제품 개선이나 사양 변경 또는 사용자 설명서 자체를 이해하기 쉽게 하기 위하여 고지 없이 변경 될 수 있습니다.
- 사용자 설명서를 손상 또는 분실해서 새로 주문할 경우에는 구입하신 대리점이나 본사로 문의해 주십시오.
- 사용자 임의로 제품을 개조하는 것은 당사의 보증 범위 밖이므로 당사에서 책임지지 않습니다.

◆ 안전 주의사항

- 설치, 운전, 점검, 보수 등을 하기 전에는 반드시 사용자 설명서를 읽어서 그 내용을 충분히 숙지하신 후에 실시 해 주십시오, 또한, 기계에 관한 지식, 안전에 관한 정보나, 주의사항을 충분히 숙지하신 후 제품을 사용하여 주십시오.
- 사용자 설명서는 안전에 관한 주의사항의 정도를 **주의**와 **경고**로 구분하여 기재하고 있습니다.

⚠**주의:**	잘못 취급했을 경우 위험한 상황을 초래하여 중상 또는 경상을 입을 가능성이 있는 경우, 그리고 대물 손해만이 발생할 가능성이 있는 경우
⚠**경고:**	잘못 취급 하였을 경우 전기 감전 등의 위험한 상황을 초래하여, 사망 또는 중상을 입을 가능성이 있는 경우

- 기재된 내용 중에 주의에 해당하는 것일지라도, 상황에 따라서 중대한 결과를 야기시킬 가능성이 있습니다. 반드시 지켜 주십시오.

◆ 제품 상태

⚠주의	제품이 손상되어 있거나 또는 부품이 빠져있는지 확인하십시오. 비정상적인 제품을 설치, 운전할 경우 기계파손 또는 부상의 위험이 있습니다.

◆ 설치

⚠주의	운전 시에는 충분히 주의하십시오 떨어지면 제품이 파손되거나, 발에 떨어지면 부상의 위험이 있습니다. 제품을 취급할 장소에는 금속 등 불연 물질을 사용하여 주십시오. 화재가 날 위험이 있습니다. 여러 대의 Ezi-MOTIONGATE를 하나의 밀폐된 공간에 설치할 때는, 냉각 장치 등을 설치하시어 주위 온도가 50℃이하가 되도록 해주십시오. 과열로 화재 또는 그 밖의 사고로 이어질 위험이 있습니다.
⚠경고	설치, 접속, 운전, 조작, 점검 및 고장 진단 작업은 적합한 자격을 가진 사람이 실시하여 주십시오. 화재, 부상, 장치 파손의 원인이 됩니다.

◆ 배선

⚠주의	드라이브의 전원 입력 전압은 정격 범위를 반드시 지켜 주십시오. 화재 및 고장의 원인이 됩니다. 접속은 배선도에 따라 확실히 실시하여 주십시오. 화재 및 오작동의 원인이 됩니다.
⚠경고	입력 전원이 OFF 되어 잇는 것을 확인한 후에 작업해 주십시오. 감전 또는 화재의 위험이 있습니다. 본 Ezi-MOTIONGATE 케이스는 콘덴서에 의해 내부회로의 GND와 절연되어 있으므로, 반드시 접지를 시켜주십시오. 감전 또는 화재의 위험이 있으며, 제품 오작동의 원인이 됩니다.

◆ 운전 및 설정 변경

⚠주의	드라이브의 보호기능이 작동하면 원인을 제거한 후에 보호 기능을 해제하여 주십시오. 원인을 제거하지 않고 운전을 계속하면 모터 및 드라이브가 오작동되어 부상, 장치 파손의 원인이 됩니다. 드라이브에 전원을 투입할 때에는 드라이브의 제어 입력을 모두 OFF로 한 후에 투입하여 주십시오. 모터가 가동되어 부상, 장치파손의 원인이 됩니다. 본 Ezi-MOTIONGATE 의 모든 값들은 출하 시 적절히 설정해 놓았습니다. 설정 변경 시에는 충분히 사용자 설명서를 숙지한 후 변경해 주십시오. 기계가 파손되거나 제품이 고장 날 수 있습니다.

◆ 보수 및 점검

⚠경고	본 Ezi-MOTIONGATE는 주 회로 전원을 차단한 후, 충분히 시간이 경과한 후에 보수, 점검을 해주십시오. 콘덴서 전원이 남아 있으므로, 감전 등의 위험이 있습니다. 통전 중에는 배선 변경을 하지 마십시오. 감전 또는 제품 파손. 기계파손 들의 위험이 있습니다. 제품의 개조는 절대로 하지 마십시오. 감전 또는 제품파손, 기계파손 등의 위험이 있으며, 해당 제품은 당사의 A/S를 받을 수 없습니다.

설치 시 주의사항.

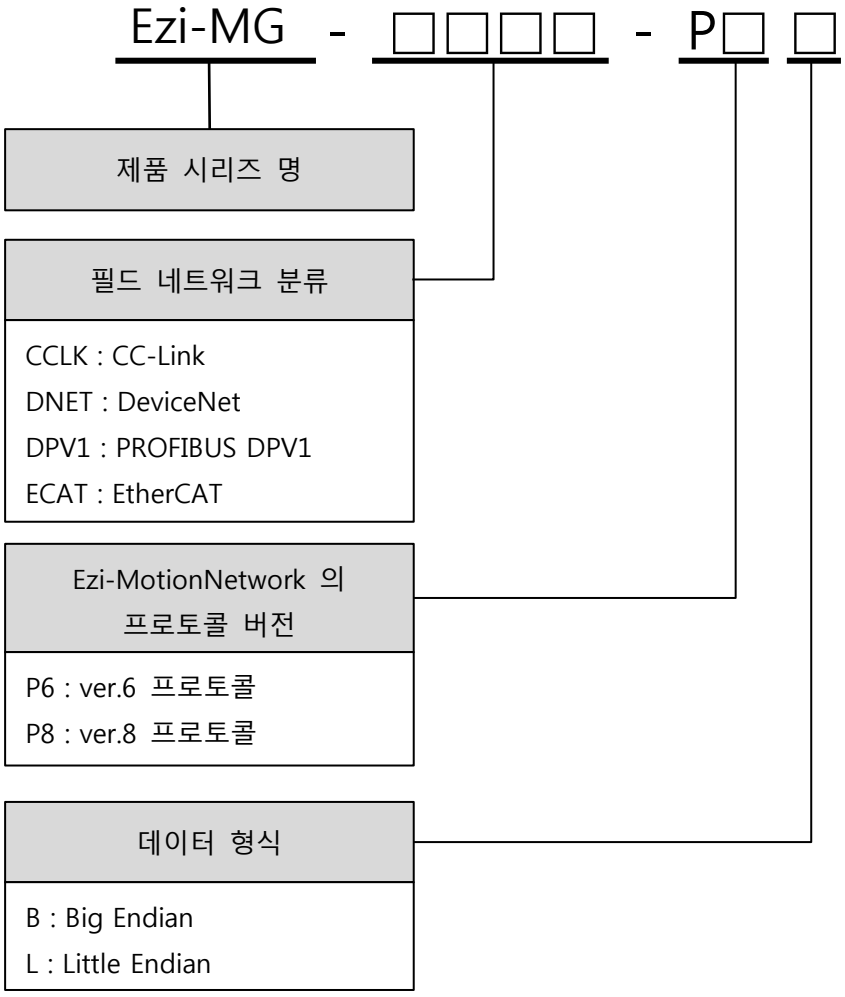
1) 실내에서 사용해야 하며, 실내 주의온도는 0°~55°C 에서 사용해야 합니다.
2) 케이스가 50°C 이상이 되면 외부에 방열을 시켜주어야 합니다.
3) 직사광선, 자석물체, 방사선 물체는 피해서 설치해 주어야 합니다.
4) 드라이브를 2대 이상 나란히 설치 시에는 수직방향은 20mm 이상, 수평방향은 50mm 이상 거리를 두고 설치해 주어야 합니다.

2. 제품 사양 및 크기

2.1 개요

- Ezi-MOTIONGATE(이하 모션게이트)는 산업용 네트워크에서 슬레이브로 연결하여, FASTECH RS485로 구성된 모션 드라이브를 마스터로 제어하는 모션 게이트웨이 장치 입니다.
- 모션게이트를 슬레이브로 연결하여 사용 가능한 가능한 최대 수량은 적용되는 산업용 네트워크에서 지원되는 수량입니다.
- 모션게이트에서 연결 가능한 모터 드라이브(Axis)는 적용되는 산업용 네트워크 마다 최대 15까지의 모터 ID를 부여할 수 있습니다

2.2 제품 형명



2.3 제품 특성 표

네트워크		EtherCAT
입력전압		24VDC ±10%
제어방식		산업용 네트워크의 I/O데이터를 사용하여 다축 제어가 가능한 모션 게이트웨이
EtherCAT 정보		EtherCAT Slave Device
물리 계층		Ethernet – 100 Base
데이터 크기		IN : 128 byte OUT : 128 byte
ECAT Device ID		스위치 설정 시 : 1~999 , 마스터 세팅 시 : 1~65535
소비전류		최대 500mA
환경	온도	사용 : 0~55℃ 보관 : -27~70℃
	습도	사용 : 35~85℃ (결로는 없을 것) 보관 : -10~90℃(결로는 없을 것)
	내 진동	0.5G
기능	스위치 선택	EtherCAT Device ID 설정
	LED 표시	네트워크 이상, 마스터 연결 이상, 드라이브의 Servo On 상태, 드라이브의 알람 상태
특수기능	조그 제어	4-Speed Step, Speed Ratio
	스텝 이동 제어	4-Step Distance
	통신 기능	Ezi-STEP Plus-R, Ezi-SERVO Plus-R series

FASTECH RS485

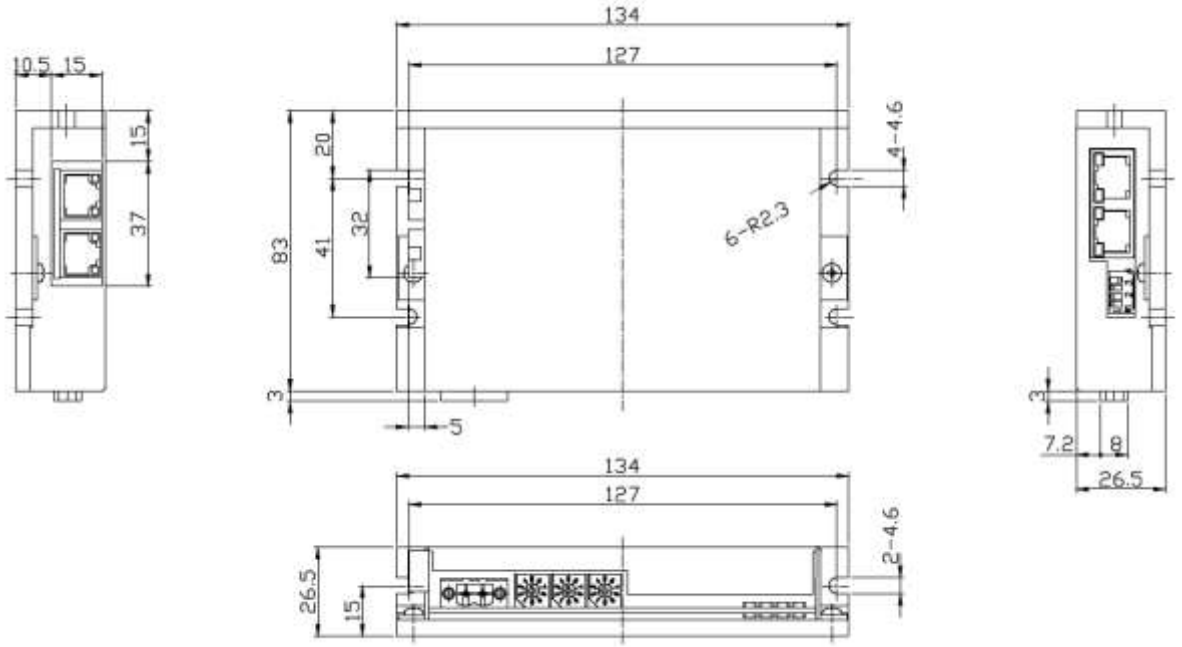
Baud-Rate (bps)	9,600	19,200	38,400	57,600	11,5200	230,400	460,800	921,600
케이블 길이 (m)	1,200	1,150	1,100	1,000	1,000	880	550	300
RJ-45 커넥터 LED	YELLOW : RS485 송신 상태 (TX from MOTIONGATE) GREEN : RS485 수신 상태 (RX to MOTIONGATE)							

NOTE 1: 케이블의 길이는 네트워크 상태가 최적일 때의 최대 연결거리입니다

NOTE 2: 산업용 네트워크의 통신규약을 준수 합니다.

2.4 제품 외형

2.4.1 제품 치수

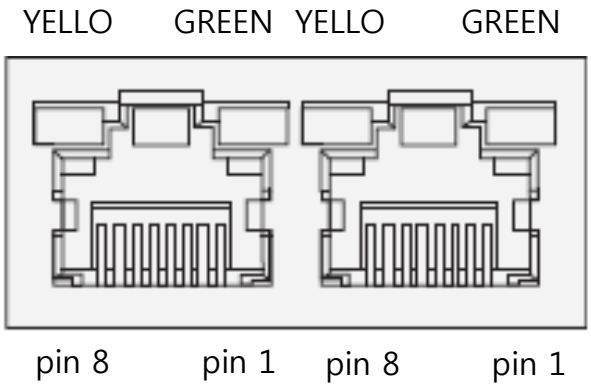


2.4.2 각 부의 명칭


통신 커넥터 (RS485)
RS485 통신속도 설정 (SW4)
드라이브의 상태 LED (LED1, LED2)
EtherCAT 상태 LED (LED3, LED4)
네트워크 국번 선택 스위치 (SW1, SW2)
네트워크 속도 선택 스위치 (SW3)
전원 커넥터 (PWR)
EtherCAT OUT
EtherCAT IN

2.4.3 FASTECH RS485 통신 접속 커넥터

통신 커넥터는 RJ-45를 사용하여 연결합니다.



통신 커넥터의 핀 맵 (RS485)

핀 번호	기능
1, 2, 4, 5, 7, 8	GND
3	DATA +
6	DATA -
CASE	Frame GND

통신 커넥터의 LED

디스플레이	색상	발광 형태
RS485 TX	녹색	RS485 데이터를 송신 시 점멸
RS485 RX	황색	RS485 데이터를 수신 시 점멸

2.4.4 RS485 통신속도 설정 및 종단 저항 선택 스위치 (SW4)

SW4 는 모터 드라이브와 연결된 RS-485 통신 네트워크의 보오레이트(Baud-Rate : 통신속도) 설정을 위한 스위치 입니다. 만약 모션게이트가 네트워크 segment 의 가장 끝 단에 연결된 상태일 경우, 종단 저항을 사용 여부를 결정할 수 있습니다.

SW4.1 은 종단 저항의 사용을 결정하며, SW4.2~SW4.4는 다음과 같이 통신속도를 설정 하는데 사용됩니다.

SW4.1	SW4.2	SW4.3	SW4.4	Speed baud[bps]
X	OFF	OFF	OFF	9600
X	ON	OFF	OFF	19200
X	OFF	ON	OFF	38400
X	ON	ON	OFF	57600
X	OFF	OFF	ON	115200 *
X	ON	OFF	ON	230400
X	OFF	ON	ON	460800
X	ON	ON	ON	921600

SW4.1가 OFF : 종단 저항이 OFF 상태.

SW4.1가 ON : 종단 저항이 ON상태.

* : 초기 설정 값

⚠주의	하나의 RS485 네트워크에 연결된 드라이브 모듈들의 통신 속도 설정 값은 모두 동일한 값으로 지정해 주어야 합니다.

2.4.5 상태 표지 LED (LED1...LED4)

상태 표시 LED는 상황에 따라 LED1~LED4를 동시에 동작 하거나, 단독적으로 동작하여 상태를 표시합니다.

LED 번호	동작 상태	설명
LED1 LED2 LED3 LED4	소등	전원 OFF, 워치-독 타임 아웃, 네트워크 미 접속
	녹색, 적색 동시 점등	모션게이트에 전원공급으로 부팅되는 상태 국번 지정 스위치 또는 네트워크 통신속도 선택 스위치를 변경 하여 재 부팅되는 상태 * 녹색과 적색이 동시 점등하므로 LED의 색상은 주황색으로 보입니다.
	녹색 동시 점멸	모션게이트의 자기 진단 상태. 커넥터 미 연결 잘못된 네트워크 보오레이트 설정 잘못된 네트워크 국번 지정
LED3 LED4	녹색, 적색 동시 점등	모션게이트의 네트워크 장치가 인식 불가 상태 * 본사 또는 대리점으로 연락하여 조치를 취하십시오.

■ **드라이브 상태 표시 LED (DRIVE STATUS)**

<table>
<tr><th>LED
번호</th><th>LED
상태 정보</th><th>LED
이름</th><th>동작
상태</th><th>설명</th></tr>
<tr><td rowspan="3">LED1</td><td rowspan="2">드라이브의
접속 상태</td><td rowspan="2">ENABLE
(녹색)</td><td>점등</td><td>CONNECT 된 드라이브 중 모터가 활성화 됨.</td></tr>
<tr><td>점멸</td><td>CONNECT 된 드라이브 중 1개 이상의 드라이브의 모터가 ENABLE 되지 않음</td></tr>
<tr><td>드라이브
알람</td><td>ALARM
(적색)</td><td>점멸</td><td>1개 이상의 드라이브가 알람상태일 때</td></tr>
<tr><td rowspan="7">LED2</td><td rowspan="2">드라이브
연결 양호</td><td>CONNECT
(녹색)</td><td>점등</td><td rowspan="2">모터 드라이브가 CONNECT 명령이 실행되어 모션게이트와 정상적으로 통신 중</td></tr>
<tr><td>ERROR
(적색)</td><td>소등</td></tr>
<tr><td rowspan="2">드라이브
접속 상태
오류</td><td>CONNECT
(녹색)</td><td>소등</td><td rowspan="2">RS-485 네트워크에 연결된 모터 드라이브가 CONNECT 명령이 실행된 모터 드라이브가 없음.
모터 드라이브와 모션게이트가 통신중인 상태가 아님.
드라이브와 통신되지 않음
RS485네트워크의 통신연결이 끊김
통신속도 설정 오류</td></tr>
<tr><td>ERROR
(적색)</td><td>점등</td></tr>
<tr><td rowspan="2">통신 에러</td><td>CONNECT
(녹색)</td><td>점등</td><td rowspan="2">모터 드라이브와 통신 오류 발생 (CRC에러 발생)</td></tr>
<tr><td>ERROR
(적색)</td><td>무작위
점멸</td></tr>
<tr><td rowspan="2">다축 연결
상태의
통신 에러</td><td>CONNECT
(녹색)</td><td>점등</td><td rowspan="2">RS485에 연결된 하나 이상의 모터드라이브에 CONNECT 명령에 응답 하지 않음.
네트워크의 연결이 끊김
토폴로지 구성에 이상이 있음.
없는 모터드라이브의 IO-Map 영역에 CONNECT 명령을 실행함</td></tr>
<tr><td>ERROR
(적색)</td><td>점등</td></tr>
</table>

⚠주의	드라이브 상태 표시 LED는 모션게이트와 드라이브의 통신 상태를 점검 후 모터 활성화 상태를 확인하도록 하십시오.

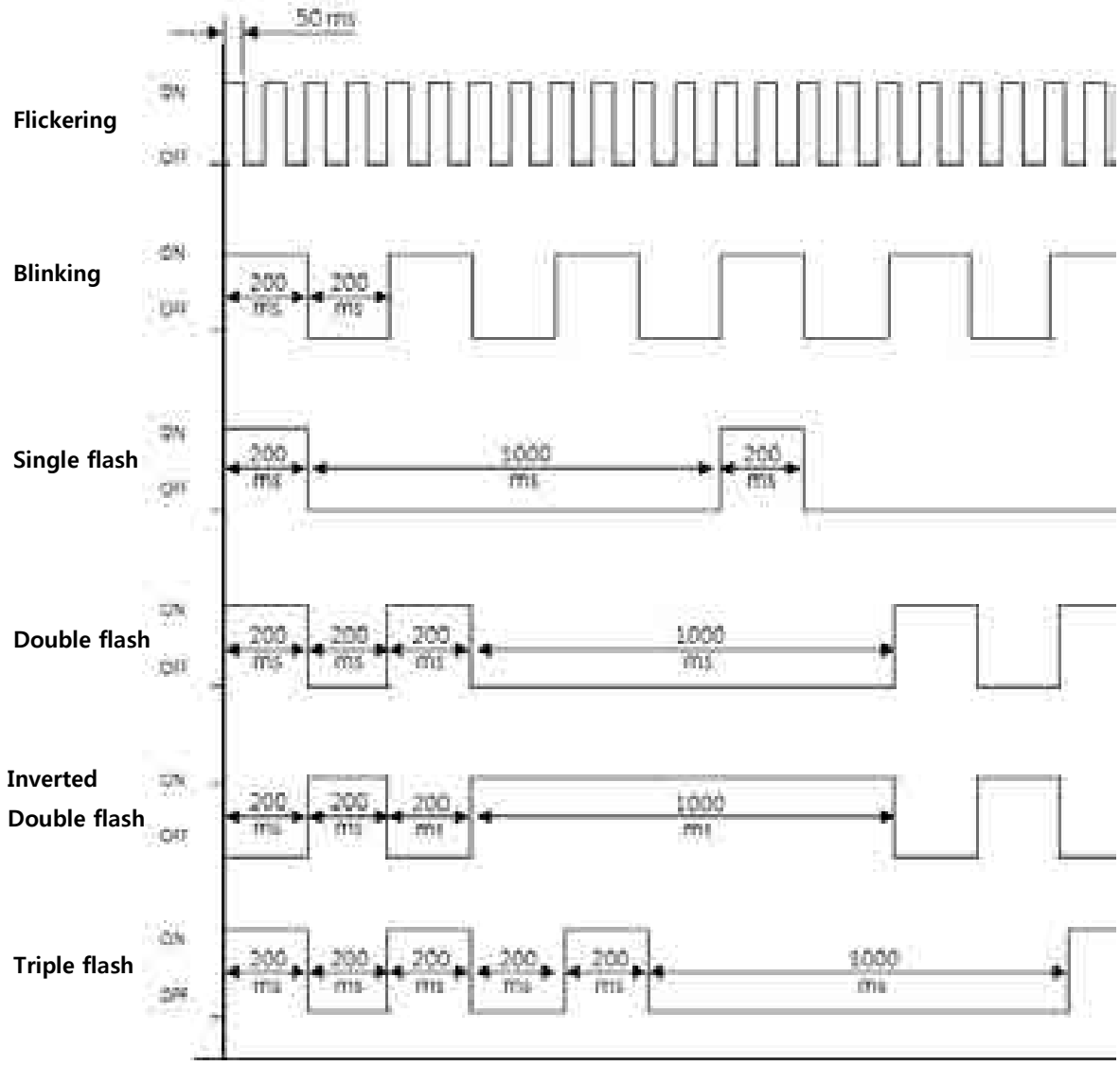
■ **EtherCAT 상태 표시 LED (EtherCAT STATUS)**

번호	LED	동작 상태	내용	설명 또는 조치
LED 3	RUN (녹색)	OFF	'INIT'-state 또는 전원 OFF	전원 상태를 확인하시오
		ON	'OPERATIONAL'-state	EtherCAT마스터와 정상 연결된 상태
		Blinking	'PRE-OPERATIONAL'-state	EtherCAT 네트워크의 초기 설정을 처리 중인 상태로 SDO 통신만 가능합니다.
		Single flash	'SAFE-OPERATIONAL'-state	SDO 통신이 가능하며, PDO 통신은 입력만 가능한 상태로, PRE-OPERATIONAL' 이후에 반드시 동작되는 상태 입니다.
	RUN (적색)	ON	치명적인 오류	모션게이트의 고장이 의심 되므로, 대리점 또는 본사에 수리의뢰를 하십시오
LED 4	ERROR (적색)	OFF	정상 동작	어떠한 에러가 발생되지 않은 상태
			전원 OFF	RUN LED의 상태가 소등 상태라면, 모션게이트의 전원을 확인 하십시오
		Single flash	설정 에러	EtherCAT 마스터의 네트워크 설정 또는 ESI파일 설정이 잘못되었습니다. - 마스터에서의 EtherCAT 슬레이브 설정을 확인하십시오.
		Double flash	워치독(WDT) 타임 아웃	Sync Manager의 타임 아웃발생 - 마스터를 점검 후 모션게이트를 re-boot 하십시오
		ON	예외적인 오류	Application controller 의 오류가 발생 - 마스터의 동작상태를 확인하십시오.

⚠주의	반드시 본사에서 제공하는 ESI 파일을 사용 하십시오. 사용자 임의로 ESI 파일을 수정 시 발생되는 문제는 책임지지 않습니다. EtherCAT Conformance Test를 실행 시, 점검 및 동작 테스트 과정에서 ERROR(적색) 표시등이 ON 되는 경우가 있으니 참고 바랍니다.

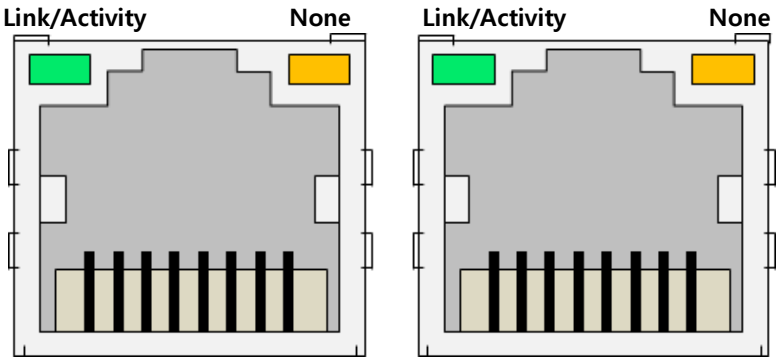
EtherCAT 의 LED 상태 표시는 육안으로 확인할 수 있도록 아래의 그림으로 표현됩니다.

Figure 1. EtherCAT LED 표시 형태



2.4.6 EtherCAT Link/Activity LED

ECAT IN 포트와 ECAT OUT 포트의 LED 는 EtherCAT Link 상태 와 활성 상태를 표시하는 Link/Activity LED 입니다. (황색 LED 는 사용되지 않습니다)



LED 상태	상태	조치
OFF	Link가 감지 되지 않음	EtherCAT Link를 감지지 않았습니다. 전원 및 연결 상태를 확인하십시오.
Green ON	EtherCAT Link를 감지 하였으나, 활성화 되지 않음	EtherCAT Link를 감지 하였으나, 트래픽을 감지하지 못하였습니다. 마스터의 운용 상태를 확인하십시오.
Green, flickering	EtherCAT Link를 감지 하였고, 활성 됨을 확인	EtherCAT Link의 감지와 트래픽을 확인 하였습니다. 정상 운용상태입니다.

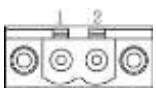
2.4.7 EtherCAT Device ID 설정 (SW1, SW2, SW3)

EtherCAT ID(ECAT Device ID)를 설정하는 로터리 스위치로, 1~999의 범위로 네트워크에서 설정하고자 하는 Device ID로 지정 할 수 있습니다. (1,000 이상의 ID를 부여할 때는 로터리 스위치 값을 0로 설정 후, 마스터에서 Write Slave ID 명령으로 지정 가능합니다.)

스위치 값 (SW3)	ID번호 X100 (100자리)	스위치 값 (SW2)	ID번호 X10 (10자리)	스위치 값 (SW1)	ID번호 X1 (1자리)	
0	000	0	00	0	0	
1	100	1	10	1	1	
2	200	2	20	2	2	
3	300	3	30	3	3	
4	400	4	40	4	4	
5	500	5	50	5	5	
6	600	6	60	6	6	
7	700	7	70	7	7	
8	800	8	80	8	8	
9	900	9	90	9	9	

2.4.8 전원 커넥터(DC POWER)

전원 공급을 위한 커넥터 입니다.

번호	기능	핀 배치도
1	입력전원 : 24VDC ± 10%	
2	입력전원 : GND	

3. 설치 및 결선 방법

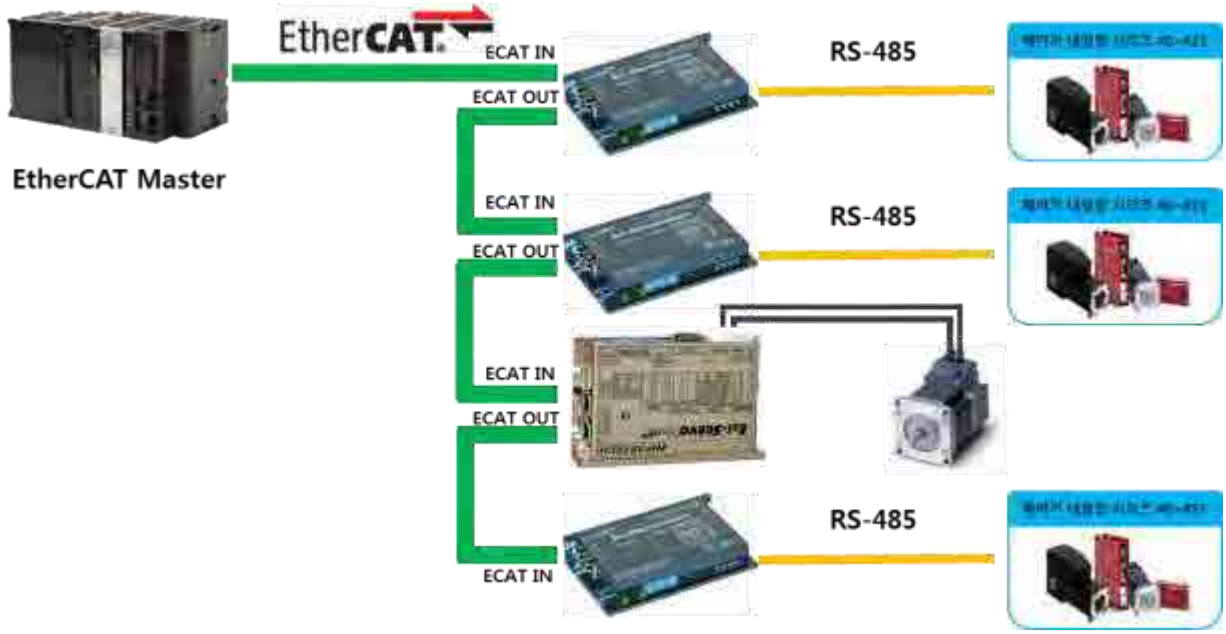
3.1 네트워크 시스템 구성도

모션게이트 EtherCAT의 네트워크는 구성은 상위제어기와 EtherCAT을 통하여 연결되고, RS485로 FASTECH 제어기 내장형 모션드라이브와 연결됩니다. 또한 EtherCAT의 토폴로지를 지원하여 다음과 같이 구성 가능합니다.

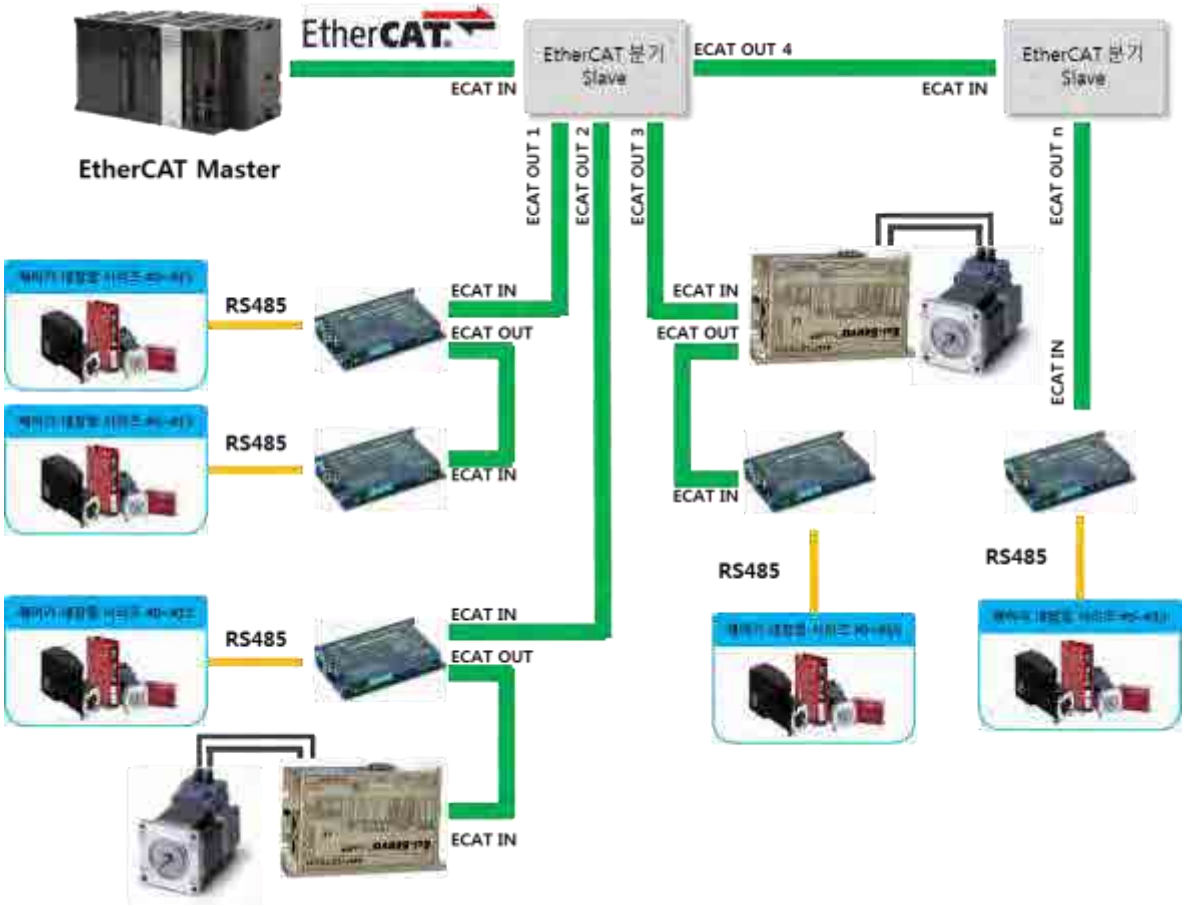
■ **모션게이트의 네트워크 배선도**

■ **EtherCAT 토폴로지 구성도**

데이지 체인 접속 방법



■ **분기 접속 방법**



⚠주의	모션게이트를 EtherCAT 네트워크에 연결 시, 마스터 또는 분기 슬레이브와 연결 되는 부분은 반드시 ECAT IN 포트로 연결 해야 하며, ECAT OUT 포트에서 다음 슬레이브 장비의 ECAT IN 포트로 연결 해야 합니다.

3.2 네트워크 결선

EtherCAT 은 물리계층이 Ethernet 기반으로 된 네트워크 입니다. RJ-45 커넥터의 사용으로 네트워크에 연결 할 수 있습니다. PC에서 사용하는 다이렉트 케이블로 사용이 가능합니다.

RJ-45 커넥터의 핀 맵(T568B)

핀 번호	데이터 명	선 색	선 색	데이터 명	핀 번호
1	TX+	흰-주황색	흰-주황색	TX+	1
2	TX-	주황색	주황색	TX-	2
3	RX+	흰-녹색	흰-녹색	RX+	3
4	Blank	청색	청색	Blank	4
5	Blank	흰-청색	흰-청색	Blank	5
6	R-	녹색	녹색	R-	6
7	Blank	흰-갈색	흰-갈색	Blank	7
8	Blank	갈색	갈색	Blank	8
커넥터 후드		케이블 실드	케이블 실드		커넥터 후드

⚠주의	모션게이트를 EtherCAT 네트워크에 연결 시 일반 UTP 케이블을 사용하여도 무관하나, 산업 현장에서의 네트워크 가설 시 노이즈에 대한 영향으로 데이터 손실이 발생할 수 있습니다. 이에 대한 대책으로 STP 케이블 및 EtherCAT 전용케이블 사용을 권장합니다

UTP/STP 케이블을 사용한 RJ45 케이블의 구조

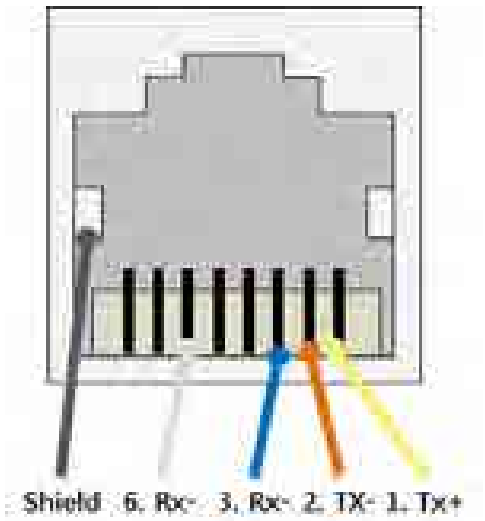
UTP/STP 케이블	RJ-45 커넥터의 핀 맵
 * 상기 그림은 STP 케이블 규격으로 UTP 케이블은 쉴드가 없는 케이블 입니다.	 * 상기 그림은 T568B의 규격입니다.

EtherCAT 네트워크 케이블의 구조

EtherCAT 케이블

심선 종류	데이터 명칭
황색	Tx+
주황색	Tx-
백색	Rx+
청색	Rx-
변조 쉴드	Shield

ECAT IN / ECAT OUT 포트의 핀 맵



* 쉴드 형 커넥터를 사용시 Shield 는 RJ-45의 커넥터에 접점 됩니다.

4. EtherCAT

4.1 EtherCAT의 개요

EtherCAT(Ethernet Control Automation Technology)은 이더넷 기반의 물리계층으로 이루어진 고성능 산업용 네트워크 입니다. 본 프로토콜은 이더넷 기술을 채택하여 EtherNet 네트워크와 동일한 케이블을 사용하여 시스템을 구축 할 수 있어 범용성이 뛰어난 네트워크 시스템 입니다. 그러나 EtherNet 프로토콜과 달리 독자적인 프로토콜 구성으로 이더넷 프레임을 빠른 속도로 전송하기 때문에 짧은 통신 사이클 타임을 실현 가능합니다. 따라서, 산업 현장에서 중소 시스템에서 시스템 통합성이 요구되는 광범위한 제어시스템까지 유연한 네트워크 구성이 가능합니다.

4.1.1 EtherCAT의 특징

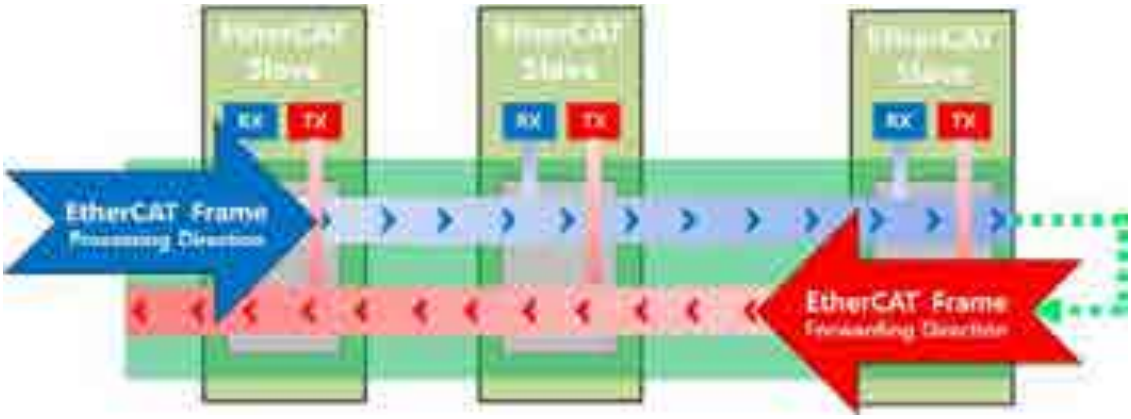
EtherCAT은 100Mbps 의 초고속 통신 속도를 지닙니다. 입력 신호에서 출력 신호까지의 응답시간을 대폭 단축 되며, Ethernet 프레임의 대역을 고속 리피트 방식으로 전송함으로써, 다양한 데이터를 높은 효율로 전송할 수 있습니다.

- **전송 속도 : 100 Mbit/s , 전-이중연결 방식(Full-Duplex)**
- **100개로 분산된 1,000 digital I/O 장비 데이터 업데이트 시간 : 30μsec**
- **유연한 토폴로지 구성**
  - 데이지 체인
  - 트리 형 토폴로지
  - 스타 형 포폴로지
  - 링 형 토폴로지

⚠주의	병합된 토폴로지 구성이 가능하나, EtherCAT 마스터에서 지원 하지 못하는 토폴로지가 있으니, 네트워크 구성 전에 EtherCAT 마스터의 제조사에 토폴로지의 지원 여부를 확인 하십시오.

4.1.2 EtherCAT의 구조

EtherCAT의 데이터는 네트워크상의 각 슬레이브 노드에 직접적으로 데이터를 송신되지 않고,, 슬레이브 노드에 이더넷 프레임을 통과됩니다. 통과 시에 슬레이브 노드에서 데이터를 판독 및 기록을 하게 됩니다. 이때, EtherCAT 마스터에서 송신된 이더넷 프레임은 슬레이브 노드에 머무르지 않고 바로 통과하여, 최종 슬레이브에 의하여 EtherCAT 마스터로 되돌려 보내지며, 다시 모든 슬레이브 노드를 통과하여 EtherCAT 마스터로 리턴 하게 됩니다. 이러한 데이터 전송 방식으로 실시간(Real-Time) 제어가 가능 합니다.

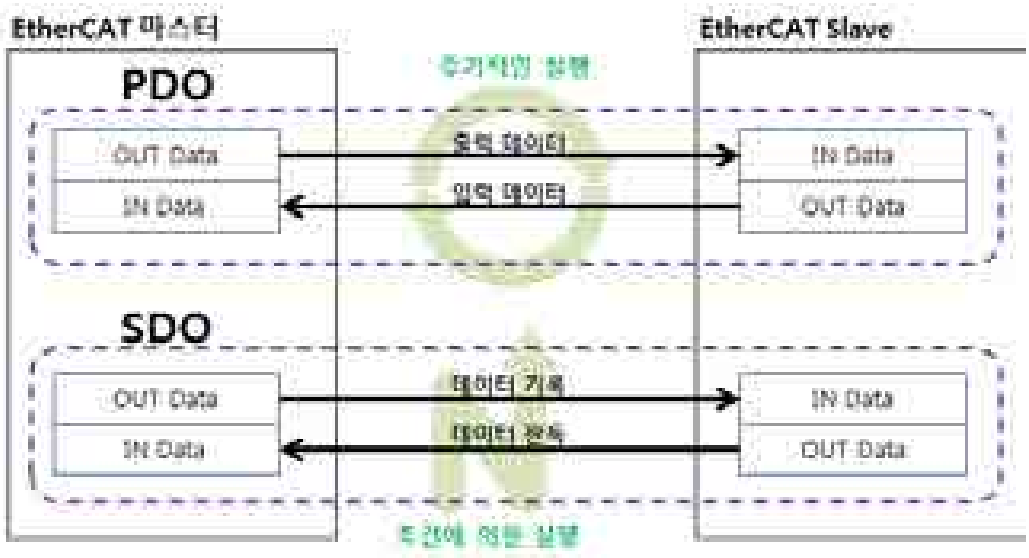


PDO (Process Data Objects)통신

일정한 주기로 실시간 정보 교환을 실행하여 정보가 갱신되는 통신

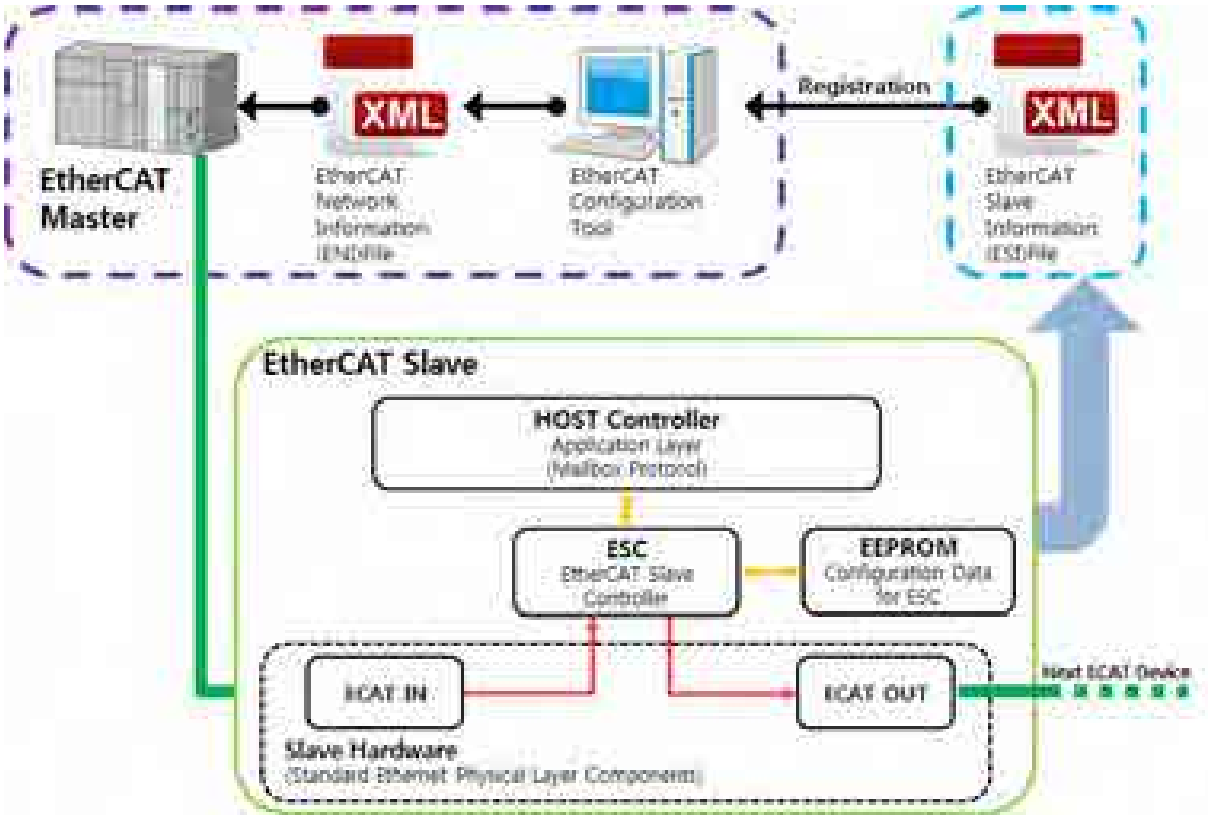
SDO (Service Data Objects) 통신

조건에 의한 실행으로 데이터를 판독 및 기록을 실행하는 통신



4.1.3 EtherCAT Slave Device의 연결

EtherCAT 의 마스터와 슬레이브 장비의 연결을 위해서는 ESI(EtherCAT Slave Information)파일이 필요합니다. ESI 파일에는 EtherCAT 사양에 준하여 개발된 슬레이브 장비의 장치 정보를 XML 형식으로 기술하는 파일입니다(제조사, 제품 유형, 제품명, IN/OUT 데이터 할당량, IN/OUT의 PDO 또는 SDO의 어드레스 및 정보 등이 기록되어 있음). ESI 파일을 EtherCAT 설정 장비를 통하여 EtherCAT 마스터 장비에 기록함으로, 슬레이브 디바이스의 PDO 및 SDO 등의 설정을 용의하게 수행 할 수 있습니다.



ECAT OUT 포트

EtherCAT 통신 프레임을 다른 기기로 전송하는 포트로, 다른 기기를 연결할 경우 반드시 ECAT IN 포트와 접속 하십시오,

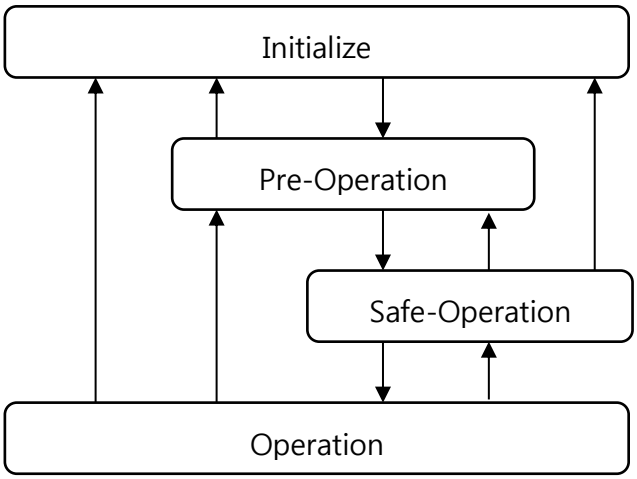
ECAT IN 포트

EtherCAT 통신 프레임이 입력되는 포트로, 반드시 다른 기기의 ECAT OUT 포트와 접속 하십시오.

⚠주의	ESI 파일은 EtherCAT 슬레이브 장비의 제조업체에서 공급받을 수 있습니다. ENI 파일과, EtherCAT 설정 툴은 EtherCAT 마스터 장비의 제조업체에서 공급 받을 수 있습니다.

4.2 EtherCAT 통신 상태와 제어 순서

EtherCAT 슬레이브의 상태 동작은 EtherCAT마스터에 의해 제어 됩니다.



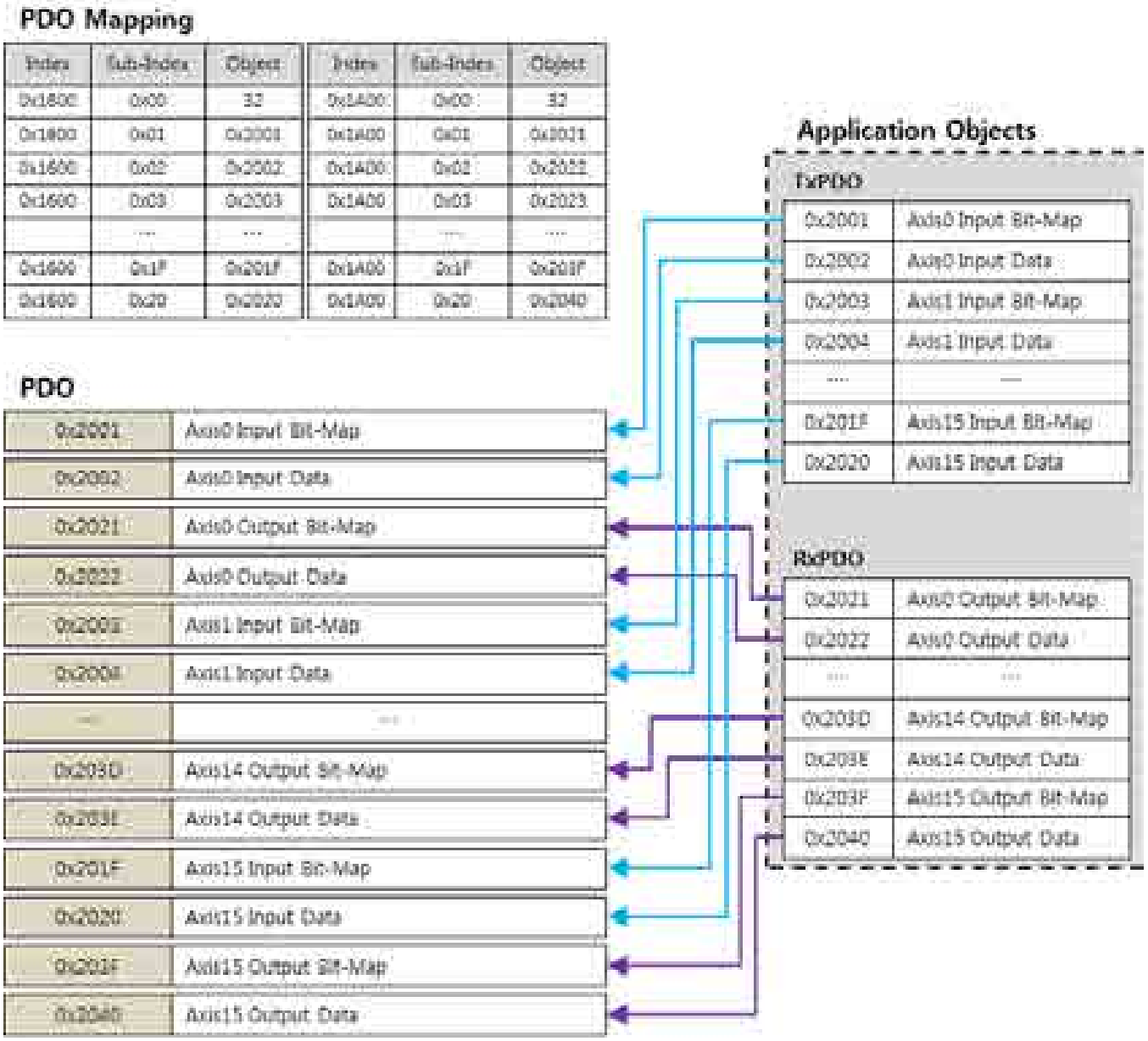
상태	SDO	RX PDO	TX PDO	내용
Initialize	불가능	불가능	불가능	통신 초기화 생태로 통신 수행을 할 수 없습니다.
Pre-Operation	가능	불가능	불가능	메일박스 통신만 가능한 상태로 최기화 이후 이 상태로 전환되어 네트워크의 초기 설정 프로세스가 실행됩니다.
Safe-Operation	가능	불가능	가능	Pre-Operation 상태에서 Operation 상태가 되기 전에 수행되는 상태입니다.
Operation	가능	가능	가능	모든 통신 조작이 가능한 상태 입니다.

4.3 PDO (프로세싱 데이터 오브젝트)

PDO 데이터는 일정한 주기로 실시간으로 정보가 교환되어 수행하게 됩니다. PDO 비트는 슬레이브에 의하여 EtherCAT 데이터 프레임에 적재 되거나, 적출되는 방식으로 수행됩니다.

PDO 맵핑 정보는 Objects dictionary의 1600h 번지에 송신 PDO 맵핑이 설정되고, 1A00h 번지에서 수신 POD 맵핑이 설정됩니다. 이 사항은 "4.5.1 Standard Objects" 설명 됩니다.

Figure 2. PDO 맵핑



4.4 서비스 데이터 객체(SDO)

SDO 는 조건에 의하여 EtherCAT 슬레이브 장비와 통신하는 명령입니다. 이 객체는 EtherCAT 슬레이브가 Pre-Operation, Safe-Operation 상태에서 명령이 가능합니다. 모션게이트는 SDO 명령 지원이 가능하며, "4.5.2 Manufacturer Specific Objects" 에서 설명하는 모션게이트의 IO-Map 에 직접 접근이 가능합니다.

4.5 EtherCAT Object Dictionary (CANopen over EtherCAT)

4.5.1 Standard Objects

Standard Object Dictionary는 DS301 통신 프로파일에 따라 구현됩니다. EtherCAT Object 와 진단 Object 의 설정은 특정 Object 항목에 해당 합니다.

Index	오브젝트 명	Sub-Index	Description	Type	Access	Notes
1000h	Device Type	00h	Device Type	U32	RO	0000 0000h (No profile)
1001h	Error register	00h		U32	RO	참조
1003h	Pre-defined error field	00h	Number of errors	U8	RW	-
		01h... 05h	Error field	U32	RO	-
1008h	device name	00h	Manufacturer device name	Visible string	RO	Ezi-MOTIONGATE
1009h	hardware version	00h	Manufacturer hardware version	Visible string	RO	EzMG_STARM_v1.1_LG5
1011h	Restore parameters	00h	Largest sub index supported	U8	RO	01h
		01h	Restore all default parameters	U32	RW	
1018h	Identity object	00h	Number of entries	U8	RO	Number of entries
		01h	Vendor ID	U32	RO	0x0FA00000
		02h	Product Code	U32	RO	0x00000010
		03h	Revision Number	U32	RO	0x00010000
		04h	Serial Number	U32	RO	0xA0126D94

4.5 EtherCAT Object Dictionary (CANopen over EtherCAT)

Index	오브젝트 명	Sub-Index	Description	Type	Access	Notes
1600h	Receive PDO mapping	00h	No. of mapped application objects in PDO	U8	RO	32 (Input-Map
		01h	Mapped object #1	U32	RO	0x2001:00, 32
		02h	Mapped object #2	U32	RO	0x2002:00, 32
		…	…	…	…	
		NNh	Mapped object #NN	U32	RO	0x2020:00, 32
1A00h	Transmit PDO mapping	00h	No. of mapped application objects in PDO	U8	RO	32
		01h	Mapped object #1	U32	RO	0x2021:00, 32
		02h	Mapped object #2	U32	RO	0x2022:00, 32
		…		…	…	
		NNh	Mapped object #NN	U32	RO	0x2040:00, 32
1C00h	Sync Manager Communication Type	00h	Number of entries	U8	RO	4
		01h	Mailbox wr	U8	RO	1
		02h	Mailbox rd	U8	RO	2
		03h	Process Data out	U8	RO	3
		04h	Process Data in	U8	RO	4
1C12h	Sync Manager Rx PDO Assign	00h	No. of assigned PDOs	U8	RO	1
		01h	Assigned PDO	U16	RO	1600h
1C13h	Sync Manager Tx PDO Assign	00h	No. of assigned PDOs	U8	RO	1
		01h	Assigned PDO	U16	RO	1A00h
1C32h	SM output parameter	00h	Number of entries	U8	RO	1
		01h	Sync mode	U16	RO	0 (FREE_RUN)
1C33h	SM input parameter	00h	Number of entries	U8	RO	1
		01h	Assigned PDO	U16	RO	0 (FREE_RUN)

■ **Object 1000h : Device Type**

장치 유형에 대한 정보가 포함 되어있는 레지스터 입니다. 모션게이트는 EtherCAT 장치 프로파일을 사용하지 않은 장치이므로 데이터는 0000h 입니다.

OBJECT DESCRIPTION

INDEX	1000h
Name	Device Type
Object Code	VAR
Data Type	UNSIGNED32

ENTRY DESCRIPTION

Access	Read Only
PDO Mapping	No
Value Range	UNSIGNED32
Default Value	No

Byte : MSB LSB

Additional Information	Device Profile Number

Figure 3 Structure of the Device Type Parameter

■ **Object 1001h : Error register**

EtherCAT 모듈에 대한 에러 레지스터입니다. 장치에 대한 에러 사항을 이 레지스터로 확인할 수 있습니다.

OBJECT DESCRIPTION

INDEX	1001h
Name	Error Register
Object Code	VAR
Data Type	UNSIGNED8
Category	Mandatory

ENTRY DESCRIPTION

Access	Read Only
PDO Mapping	no
Value Range	UNSIGNED8
Default Value	No

에러 레지스터의 비트가 1 로 세트 되었을 때 해당 에러가 발생됨을 확인할 수 있습니다. 0 번 비트의 에러는 어떠한 에러가 발생하였을 때 표시가 되는 비트 입니다.

bit	M/O	의미
0	M	에러 발생
1	O	전류
2	O	전압
3	O	온도
4	O	통신 에러
5	O	장치 프로파일 특정 오류
6	O	사용하지 않음(항상 0)
7	O	제조사의 특정 오류

■ **Object 1003h : Pre-defined error field**

에러가 발생하였을 때 이 영역에 발생된 오류를 지니며, 긴급 오브젝트를 통하여 신호를 보냅니다. 이러한 동작을 수행한 에러 기록을 제공합니다.

① 발생된 에러는 서브 인덱스 1 부터 시작하여 기록됩니다.

② 새로 발생된 에러는 서브 인덱스 1 부터 기록되며, 기존의 기록된 정보는 다음 서브 인덱스 번호로 이동합니다. (stack: 선형 자료 구조)

③ 에러 내역이 "0"개일 때, 서브 인덱스 0 보다 큰 위치의 서브 인덱스 영역에 기록될 수 없습니다.

④ 에러 메시지는 16 비트의 발생된 오류 코드(LSB)와 16 비트의 정보 코드 (MSB)를 포함하여 UNSIGNED32 형식으로 되어있습니다.

Byte : MSB LSB

Additional Information	Error Code

Figure 4 Structure of the pre-defined error field

에러 코드 표

Error Code (hex)	에러 메시지
00xx	Error Reset or No Error
10xx	Generic Error
20xx	Current
21xx	Current, device input side
22xx	Current inside the device
23xx	Current, device output side
30xx	Voltage
31xx	Mains Voltage
32xx	Voltage inside the device
33xx	Output Voltage
40xx	Temperature
41xx	Ambient Temperature
42xx	Device Temperature

Error Code (hex)	에러 메시지
50xx	Device Hardware
60xx	Device Software
61xx	Internal Software
62xx	User Software
63xx	Data Set
70xx	Additional Modules
80xx	Monitoring
81xx	Communication
8110	CAN Overrun (Objects lost)
8120	CAN in Error Passive Mode
8130	Life Guard Error or Heartbeat Error
8140	recovered from bus off
8150	Transmit COB-ID collision
82xx	Protocol Error
8210	PDO not processed due to length error
8220	PDO length exceeded
90xx	External Error
F0xx	Additional Functions
FFxx	Device specific

OBJECT DESCRIPTION

INDEX	1003h
Name	pre-defined error field
Object Code	ARRAY
Data Type	UNSIGNED32

ENTRY DESCRIPTION

Sub-Index	0h
Description	number of errors
Access	Read Only
PDO Mapping	No
Value Range	0~254
Default Value	0

Sub-Index	1h
Description	Standard error field
Access	Read Only
PDO Mapping	No
Value Range	UNSIGNED32
Default Value	No

Sub-Index	2h~FEh
Description	Standard error field
Access	Read Only
PDO Mapping	No
Value Range	UNSIGNED32
Default Value	No

■ **Object 1008h : device name**

장치의 이름이 명시되어 있습니다. 모션게이트에는 Ezi-MOTIONGATE 로 표기 됩니다.

OBJECT DESCRIPTION

INDEX	1008h
Name	Device Name
Object Code	VAR
Data Type	Visible String

ENTRY DESCRIPTION

Access	Read Only
PDO Mapping	No
Value Range	No
Default Value	No

■ **Object 1009h : hardware version**

모션게이트의 하드웨어 버전이 명시되어 있습니다.

OBJECT DESCRIPTION

INDEX	1009h
Name	Hardware Version
Object Code	VAR
Data Type	Visible String

ENTRY DESCRIPTION

Access	Read Only
PDO Mapping	No
Value Range	No
Default Value	No

■ **Object 1011h : Restore parameters**

통신 또는 장치 프로파일에 따라서 파라미터가 이 오브젝트의 기본값으로 복원됩니다. 복원될 값은 서브 오브젝트 01h 의 값으로 복원됩니다.

서브-인덱스 0 : 지원되는 최대 규모의 서브-인덱스를 포함
서브-인덱스 1 : 복원할 수 있는 모든 파라미터
서브-인덱스 2 : 통신 관련 매개 변수
서브-인덱스 3 : 응용프로그램 관련 매개 변수

사용자가 실수로 기본 파라미터의 복원하게 되는 것을 방지하기 위하여, 해당 서브 인덱스에 특정 서명이 작성되었을 때만 복원됩니다. 여기서 특정서명은 "load"입니다.

Signature	MSB			LSB
ASCII	D	a	o	l
hex	64h	61h	6Fh	6Ch

Figure 5 Restoring write access signature

SDO 전송 확인을 해당 서브-인덱스의 올바른 서명의 리셉션에 장치가 기본 매개 변수를 복원한 후 실행합니다(다운로드 응답을 시작). 복원이 실패한 경우, 장치가 중지 SDO 의 전송으로 응답 합니다(중지코드 : 0606 0000h). 잘못된 서명이 작성되었을 경우, 장치가 기본값을 복원을 거부하고 중지 SDO 의 전송으로 응답합니다(중지코드 : 0800 002xh).

기본값은 장치의 리셋(서브-인덱스 1h~7Fh 에 대한 리셋 로드, 서브-인덱스 2 에 대한 리셋 통신) 또는 전원 재 공급 후에 유효한 값으로 설정 할 수 있습니다.

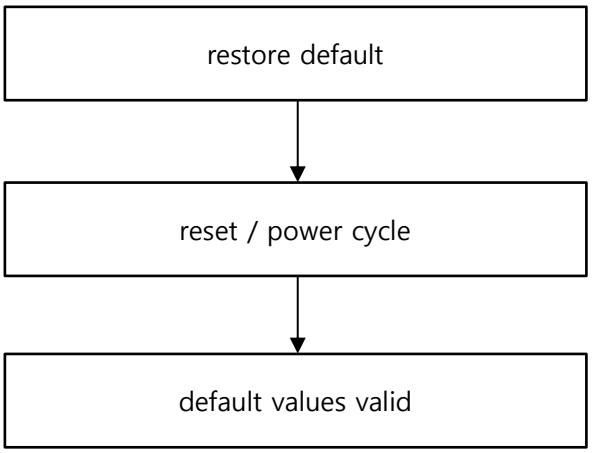


Figure 6 restore procedure

해당 서브-인덱스에 대한 읽기 권한은 Figure 5 와 같은 형식으로 기본 파라미터 복원기능에 대한 정보를 제공합니다.

UNSIGNED 32

bits	MSB	LSB
	31~1	0
	reserved (=0)	0/1

Figure 7 Restoring default values read access structure

bit	Value	의미
0	0	장치가 기본 파라미터를 복원 하지 않음
	1	장치가 기본 파라미터를 복원
31~1	0	reserved (=0)

OBJECT DESCRIPTION

INDEX	1011h
Name	restore default parameters
Object Code	ARRAY
Data Type	UNSIGNED32

ENTRY DESCRIPTION

Sub-Index	0h
Description	largest sub-index supported
Access	Read Only
PDO Mapping	No
Value Range	1h~7Fh
Default Value	No

Sub-Index	1h
Description	restore all default parameters
Access	Read / Write
PDO Mapping	No
Value Range	UNSIGNED32 (Figure 5)
Default Value	No

Sub-Index	2h
Description	restore communication default parameters
Access	Read / Write
PDO Mapping	No
Value Range	UNSIGNED32 (Figure 5)
Default Value	No

Sub-Index	3h
Description	restore application default parameters
Access	Read / Write
PDO Mapping	No
Value Range	UNSIGNED32 (Figure 5)
Default Value	No

Sub-Index	4~7Fh
Description	restore manufacturer defined default parameters
PDO Mapping	No
Value Range	UNSIGNED32 (Figure 5)

■ **Object 1018h : Identity object**

장치에 대한 일반 정보가 기록된 오브젝트 입니다. Vender ID 와 Product code, Revision 정보, 시리얼 번호가 포함 되어있습니다.

OBJECT DESCRIPTION

INDEX	1018h
Name	Identity Object
Object Code	RECORD
Data Type	Identity

ENTRY DESCRIPTION

Sub-Index	0h
Description	number of entries
Access	Read Only
PDO Mapping	No
Value Range	1…4
Default Value	No

Sub-Index	1h
Description	Vendor ID
Access	Read Only
PDO Mapping	No
Value Range	UNSIGNED32
Default Value	No

Sub-Index	2h
Description	Product code
Access	Read Only
PDO Mapping	No
Value Range	UNSIGNED32
Default Value	No

Sub-Index	3h
Description	Revision number
Access	Read Only
PDO Mapping	No
Value Range	UNSIGNED32
Default Value	No

Sub-Index	4h
Description	Serial number
Access	Read Only
PDO Mapping	No
Value Range	UNSIGNED32
Default Value	No

■ **Object 1600h : Receive PDO mapping**

EtherCAT 슬레이브 장치가 수신 할 수 있는 PDO 에 대한 맵핑을 포함합니다.

오브젝트의 스트럭쳐의 구조는 Figure 7 과 같으며, 서브-인덱스 1h~40h 는 이 구조와 같습니다.

Byte : MSB LSB

index (16 bit)	sub-index (8 bit)	object length (8 bit)

Figure 8 Structure of PDO Mapping Entry

OBJECT DESCRIPTION

INDEX	1600h – 17FFh
Name	receive PDO mapping
Object Code	RECORD
Data Type	PDO Mapping

ENTRY DESCRIPTION

Sub-Index	0h
Description	number of mapped application objects in PDO
Access	Read Only
PDO Mapping	No
Value Range	0: deactivated 1 – 64: activated
Default Value	(device profile dependent)

Sub-Index	1h ~ 40h
Description	PDO mapping for the nth application object to be mapped
Access	Read / Write
PDO Mapping	No
Value Range	UNSIGNED32
Default Value	(device profile dependent)

인덱스	서브-인덱스	오브젝트	오브젝트 명
0x1600	0x00	32	
	0x01	0x2001	Axis0 Input Bit-Map
	0x02	0x2002	Axis0 Input Data
	0x03	0x2003	Axis1 Input Bit-Map
	0x04	0x2004	Axis1 Input Data
	0x05	0x2005	Axis2 Input Bit-Map
	0x06	0x2006	Axis2 Input Data
	0x07	0x2007	Axis3 Input Bit-Map
	0x08	0x2008	Axis3 Input Data
	0x09	0x2009	Axis4 Input Bit-Map
	0x0A	0x200A	Axis4 Input Data
	0x0B	0x200B	Axis5 Input Bit-Map
	0x0C	0x200C	Axis5 Input Data
	0x0D	0x200D	Axis6 Input Bit-Map
	0x0E	0x200E	Axis6 Input Data
	0x0F	0x200F	Axis7 Input Bit-Map
	0x10	0x2010	Axis7 Input Data
	0x11	0x2011	Axis8 Input Bit-Map
	0x12	0x2012	Axis8 Input Data
	0x13	0x2013	Axis9 Input Bit-Map
	0x14	0x2014	Axis9 Input Data
	0x15	0x2015	Axis10 Input Bit-Map
	0x16	0x2016	Axis10 Input Data
	0x17	0x2017	Axis11 Input Bit-Map
	0x18	0x2018	Axis11 Input Data
	0x19	0x2019	Axis12 Input Bit-Map
	0x1A	0x201A	Axis12 Input Data
	0x1B	0x201B	Axis13 Input Bit-Map
	0x1C	0x201C	Axis13 Input Data
	0x1D	0x201D	Axis14 Input Bit-Map
	0x1E	0x201E	Axis14 Input Data
	0x1F	0x201F	Axis15 Input Bit-Map
	0x20	0x2020	Axis15 Input Data

■ **Object 1A00h : Transmit PDO mapping**

EtherCAT 슬레이브 장치가 송신 할 수 있는 PDO 에 대한 맵핑 입니다.

오브젝트의 스트럭쳐의 구조는 Figure 7 과 같으며, 서브-인덱스 1h~40h 는 이 구조와 같습니다(Object 1600h Receive PDO mapping 의 구조와 같습니다)

Byte : MSB		LSB
index (16 bit)	sub-index (8 bit)	object length (8 bit)

Figure 9 Structure of PDO Mapping Entry

OBJECT DESCRIPTION

INDEX	1A00h – 1BFFh
Name	transmit PDO mapping
Object Code	RECORD
Data Type	PDO Mapping

ENTRY DESCRIPTION

Sub-Index	0h
Description	number of mapped application objects in PDO
Access	Read Only; Read / Write (if dynamic mapping is supported)
PDO Mapping	No
Value Range	0: deactivated 1 – 64: activated
Default Value	(device profile dependent)

Sub-Index	1h ~ 40h
Description	PDO mapping for the nth application object to be mapped
Access	Read / Write
PDO Mapping	No
Value Range	UNSIGNED32
Default Value	(device profile dependent)

인덱스	서브-인덱스	오브젝트	오브젝트 명
0x1A00	0x00	32	
	0x01	0x2021	Axis0 Output Bit-Map
	0x02	0x2022	Axis0 Output Data
	0x03	0x2023	Axis1 Output Bit-Map
	0x04	0x2024	Axis1 Output Data
	0x05	0x2025	Axis2 Output Bit-Map
	0x06	0x2026	Axis2 Output Data
	0x07	0x2027	Axis3 Output Bit-Map
	0x08	0x2028	Axis3 Output Data
	0x09	0x2029	Axis4 Output Bit-Map
	0x0A	0x202A	Axis4 Output Data
	0x0B	0x202B	Axis5 Output Bit-Map
	0x0C	0x202C	Axis5 Output Data
	0x0D	0x202D	Axis6 Output Bit-Map
	0x0E	0x202E	Axis6 Output Data
	0x0F	0x202F	Axis7 Output Bit-Map
	0x10	0x2030	Axis7 Output Data
	0x11	0x2031	Axis8 Output Bit-Map
	0x12	0x2032	Axis8 Output Data
	0x13	0x2033	Axis9 Output Bit-Map
	0x14	0x2034	Axis9 Output Data
	0x15	0x2035	Axis10 Output Bit-Map
	0x16	0x2036	Axis10 Output Data
	0x17	0x2037	Axis11 Output Bit-Map
	0x18	0x2038	Axis11 Output Data
	0x19	0x2039	Axis12 Output Bit-Map
	0x1A	0x203A	Axis12 Output Data
	0x1B	0x203B	Axis13 Output Bit-Map
	0x1C	0x203C	Axis13 Output Data
	0x1D	0x203D	Axis14 Output Bit-Map
	0x1E	0x203E	Axis14 Output Data
	0x1F	0x203F	Axis15 Output Bit-Map
	0x20	0x2040	Axis15 Output Data

■ **Object 1C00h : Sync Manager Communication Type**

Sync Manager 는 EtherCAT 마스터와 슬레이브 장치간의 안정적인 데이터 교환을 위하여 설정의 변화 시 인터럽트를 제공합니다. Sync Manager 는 EtherCAT 마스터에서 통신 모드와 통신 방향을 구성할 수 있습니다. 그리고 데이터를 교환하기 위한 메모리 영역의 버퍼 위치를 사용하며, 이 버퍼에 대한 엑세스는 Sync Manager 의 하드웨어 의해 제어됩니다.

OBJECT DESCRIPTION

INDEX	1C00h
Name	Sync Manager Communication Type
Object Code	-
Data Type	-

ENTRY DESCRIPTION

Sub-Index	0h
Description	Number of used Sync Manager channels
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	4

Sub-Index	1h
Description	Communication Type Sync Manager 0
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	0

Sub-Index	2h
Description	Communication Type Sync Manager 1
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	1

Sub-Index	3h
Description	Communication Type Sync Manager 2
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	2

Sub-Index	4h
Description	Communication Type Sync Manager 3
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	3

■ **Object 1C12 : Sync Manager Rx PDO Assign**

Sync Manager 에서 사용하는 수신 PDO 의 할당영역을 나타냅니다.

OBJECT DESCRIPTION

INDEX	1C12h
Name	Sync Manager Rx PDO Assign
Object Code	VAR
Data Type	-

ENTRY DESCRIPTION

Sub-Index	0h
Description	Number of used Sync Manager channels
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	1

Sub-Index	1h
Description	Assigned PDO
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	1600h

■ **Object 1C13h : Sync Manager Tx PDO Assign**

Sync Manager 에서 사용하는 송신 PDO 의 할당영역을 나타냅니다.

OBJECT DESCRIPTION

INDEX	1C13h
Name	Sync Manager Tx PDO Assign
Object Code	VAR
Data Type	-

ENTRY DESCRIPTION

Sub-Index	0h
Description	Number of used Sync Manager channels
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	1

Sub-Index	1h
Description	Assigned PDO
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	1A00h

■ **Object 1C32h : SM output parameter**

모션게이트의 Sync 모드는 Free Run 모드로 사용합니다.

OBJECT DESCRIPTION

INDEX	1C32h
Name	SM output parameter
Object Code	VAR
Data Type	UNSIGNED8

ENTRY DESCRIPTION

Sub-Index	0h
Description	Number of entries
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	1

Sub-Index	1h
Description	Sync mode
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	0 (FREE_RUN)

■ **Object 1C33h : SM input parameter**

모션게이트의 Sync 모드는 Free Run 모드로 사용합니다.

OBJECT DESCRIPTION

INDEX	1C33h
Name	SM input parameter
Object Code	VAR
Data Type	UNSIGNED8

ENTRY DESCRIPTION

Sub-Index	0h
Description	Number of entries
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	1

Sub-Index	1h
Description	Assigned PDO
Access	Read Only
PDO Mapping	No
Value Range	-
Default Value	0 (FREE_RUN)

4.5.2 Manufacturer Specific Objects

모션게이트의 데이터는 제조사 특정 범위의 오브젝트(2001h~5FFFh)을 사용합니다. 모션게이트에서 사용되는 오브젝트는 각축에 대한 Input Bit-Map, Input Data, Output Bit-Map, Output Data로 구분됩니다.

Index	Object Name	Sub-Index	Type	Access	Notes
2001h	Axis0 Input Bit-Map	00h~03h	DWORD	RO	
2002h	Axis0 Input Data	00h~03h	DINT	RO	
2003h	Axis1 Input Bit-Map	00h~03h	DWORD	RO	
2004h	Axis1 Input Data	00h~03h	DINT	RO	
2005h	Axis2 Input Bit-Map	00h~03h	DWORD	RO	
2006h	Axis2 Input Data	00h~03h	DINT	RO	
2007h	Axis3 Input Bit-Map	00h~03h	DWORD	RO	
2008h	Axis3 Input Data	00h~03h	DINT	RO	
2009h	Axis4 Input Bit-Map	00h~03h	DWORD	RO	
200Ah	Axis4 Input Data	00h~03h	DINT	RO	
200Bh	Axis5 Input Bit-Map	00h~03h	DWORD	RO	
200Ch	Axis5 Input Data	00h~03h	DINT	RO	
200Dh	Axis6 Input Bit-Map	00h~03h	DWORD	RO	
200Eh	Axis6 Input Data	00h~03h	DINT	RO	
200Fh	Axis7 Input Bit-Map	00h~03h	DWORD	RO	
2010h	Axis7 Input Data	00h~03h	DINT	RO	
2011h	Axis8 Input Bit-Map	00h~03h	DWORD	RO	
2012h	Axis8 Input Data	00h~03h	DINT	RO	
2013h	Axis9 Input Bit-Map	00h~03h	DWORD	RO	
2014h	Axis9 Input Data	00h~03h	DINT	RO	
2015h	Axis10 Input Bit-Map	00h~03h	DWORD	RO	
2016h	Axis10 Input Data	00h~03h	DINT	RO	
2017h	Axis11 Input Bit-Map	00h~03h	DWORD	RO	
2018h	Axis11 Input Data	00h~03h	DINT	RO	
2019h	Axis12 Input Bit-Map	00h~03h	DWORD	RO	
201Ah	Axis12 Input Data	00h~03h	DINT	RO	
201Bh	Axis13 Input Bit-Map	00h~03h	DWORD	RO	
201Ch	Axis13 Input Data	00h~03h	DINT	RO	
201Dh	Axis14 Input Bit-Map	00h~03h	DWORD	RO	
201Eh	Axis14 Input Data	00h~03h	DINT	RO	
201Fh	Axis15 Input Bit-Map	00h~03h	DWORD	RO	
2020h	Axis15 Input Data	00h~03h	DINT	RO	

Index	Object Name	Sub-Index	Type	Access	Notes
2021h	Axis0 Output Bit-Map	00h~03h	DWORD	RW	
2022h	Axis0 Output Data	00h~03h	DINT	RW	
2023h	Axis1 Output Bit-Map	00h~03h	DWORD	RW	
2024h	Axis1 Output Data	00h~03h	DINT	RW	
2025h	Axis2 Output Bit-Map	00h~03h	DWORD	RW	
2026h	Axis2 Output Data	00h~03h	DINT	RW	
2027h	Axis3 Output Bit-Map	00h~03h	DWORD	RW	
2028h	Axis3 Output Data	00h~03h	DINT	RW	
2029h	Axis4 Output Bit-Map	00h~03h	DWORD	RW	
202Ah	Axis4 Output Data	00h~03h	DINT	RW	
202Bh	Axis5 Output Bit-Map	00h~03h	DWORD	RW	
202Ch	Axis5 Output Data	00h~03h	DINT	RW	
202Dh	Axis6 Output Bit-Map	00h~03h	DWORD	RW	
202Eh	Axis6 Output Data	00h~03h	DINT	RW	
202Fh	Axis7 Output Bit-Map	00h~03h	DWORD	RW	
2030h	Axis7 Output Data	00h~03h	DINT	RW	
2031h	Axis8 Output Bit-Map	00h~03h	DWORD	RW	
2032h	Axis8 Output Data	00h~03h	DWORD	RW	
2033h	Axis9 Output Bit-Map	00h~03h	DINT	RW	
2034h	Axis9 Output Data	00h~03h	DWORD	RW	
2035h	Axis10 Output Bit-Map	00h~03h	DINT	RW	
2036h	Axis10 Output Data	00h~03h	DWORD	RW	
2037h	Axis11 Output Bit-Map	00h~03h	DINT	RW	
2038h	Axis11 Output Data	00h~03h	DWORD	RW	
2039h	Axis12 Output Bit-Map	00h~03h	DINT	RW	
203Ah	Axis12 Output Data	00h~03h	DWORD	RW	
203Bh	Axis13 Output Bit-Map	00h~03h	DINT	RW	
203Ch	Axis13 Output Data	00h~03h	DWORD	RW	
203Dh	Axis14 Output Bit-Map	00h~03h	DINT	RW	
203Eh	Axis14 Output Data	00h~03h	DWORD	RW	
203Fh	Axis15 Output Bit-Map	00h~03h	DINT	RW	
2030h	Axis15 Output Data	00h~03h	DWORD	RW	

■ **Object 2001h...2010Fh : Axis*n* Input Bit-Map**

모션게이트에 연결된 드라이브의 모션 또는 세팅제어를 위한 비트맵 영역입니다.

DWORD	MSB			LSB
Byte	4 byte area	2 byte area	1 byte area	0 byte area
Motion Mode	모션 제어	기본 모션 제어	명령선택, 응답설정	기본 제어
Setting Mode	INDEX area		명령선택	

OBJECT DESCRIPTION

INDEX	2001h… 201Fh
Name	Axis*n* Input Bit-Map
Object Code	VAR
Data Type	DWORD

ENTRY DESCRIPTION

Access	Read / Write
PDO Mapping	Yes
Value Range	DWORD
Default Value	No

Input Bit-Map 의 비트 정보는 다음과 같습니다. 아래의 설명은 비트가 1 로 세트 되었을 때의 설명입니다.

Table 1 Input Bit-Map의 비트 정보

Bit	Name	의미
0	CONNECT	모션게이트와 모션 드라이브의 통신을 연결합니다.
1	ENABLE	드라이브의 모터를 활성화 합니다.
2	/ESTOP	운전중인 모터를 비상정지 합니다.
3	ALM_RST	드라이브의 알람상태를 해지합니다. (STEP 드라이브일 경우 Motor Free)
4	CMD_START	명령 실행 비트 입니다.
5	-	
6	-	
7	MOTION/SETTING	IO-Map 의 유형을 변경 합니다.
8~11	CMD_CODE	명령의 종류를 선택 합니다.
12~15	RESPONSE _TYPE	응답 데이터의 종류를 선택 합니다.
16	CANCLE	모션 명령에 대한 취소 명령입니다.
17	HOLD	모션 중 일시 정지 명령을 실행합니다.
18	-	
19	GO_ZERO_POS	영점 이동 명령을 실행합니다.
20	JOG_MOVE -	조그 + 명령을 실행합니다.
21	JOG_MOVE +	조그 – 명령을 실행합니다.
22	STEP_MOVE-	스텝 + 이동 명령을 실행합니다.
23	STEP_MOVE+	스텝 – 이동명령을 실행합니다.
24	INC/ABS	위치이동 명령의 유형을 선택합니다.
25	-	
26	SPD_MODE	조그운전 명령의 유형을 선택합니다.
27	-	
28	SINGLE_PT	단일 PT 운전 명령을 선택합니다.
29	-	
30	-	
31	-	

Input Bit-Map 구조의 설명은 "5.3 IO-Map 의 구조" 항 에서 설명합니다.

■ **Object 2002h...2020h : Axis*n* Input Data**

모션게이트에 연결된 드라이브의 제어를 위한 데이터 영역입니다.

DINT	MSB			LSB
Byte	4 byte area	2 byte area	1 byte area	0 byte area
	32bit Data			

OBJECT DESCRIPTION

INDEX	2002h… 2020h
Name	Axis*n* Input Data
Object Code	VAR
Data Type	DINT

ENTRY DESCRIPTION

Access	Read / Write
PDO Mapping	Yes
Value Range	DINT
Default Value	No

■ **Object 2021h…202Fh : Axis*n* Output Bit-Map**

모션게이트에 연결된 드라이브의 모션 또는 세팅제어에 대한 응답 비트맵 영역입니다.

DWORD	MSB			LSB
Byte	4 byte area	2 byte area	1 byte area	0 byte area
Motion Mode	드라이브의 상태비트	기본 모션 제어의 응답 비트	명령선택, 응답설정의 응답 비트	기본 제어의 응답비트
Setting Mode	INDEX 응답 영역		명령선택의 응답 비트	

OBJECT DESCRIPTION

INDEX	2021h… 202Fh
Name	Axis*n* Input Bit-Map
Object Code	VAR
Data Type	DWORD

ENTRY DESCRIPTION

Access	Read Only
PDO Mapping	Yes
Value Range	DWORD
Default Value	0xFFFE(Hex), -2(dec) : 최초의 전원을 인가 후 Input Bit-Map 의 0 bit 가 0 일 때

Output Bit-Map 의 비트 정보는 다음과 같습니다. 아래의 설명은 비트가 1 로 세트 되었을 때의 설명입니다.

Table 2 Output Bit-Map의 비트 정보

bit	Name	설명
0	CONNECTED	드라이브와 연결 되었습니다
1	ENABLED	드라이브의 모터가 활성화 되었습니다.
2	ESTOPED	드라이브의 모터가 비상정지 상태입니다.
3	ALM/ERROR	드라이브의 알람 또는 에러가 발생하였습니다.
4	CMD_START_RESP	명령 시작 명령에 대한 응답 비트입니다.
5	-	
6	-	
7	MOTION/SETTING_RESP	현재의 IO-Map 의 유형에 대한 상태 비트입니다.
8~11	CMD_CODE_RESP	명령 종류의 응답 비트입니다.
12~15	RESPONSE_TYPE_RESP	응답 데이터의 종류에 대한 응답 비트입니다.
16	MOTIONING	드라이브의 모터가 구동 상태입니다.
17	PAUSE	드라이브의 모터가 일시 정지 상태입니다.
18	-	
19	GO_ZERO_POS_RESP	영점 이동명령의 응답 비트입니다.
20		
21	JOG_MOVE_RESP	조그운전 명령의 응답 비트입니다.
22		
23	STEP_MOVE_RESP	스텝이동 명열의 응답 비트입니다.
24	PT_RUNING	PT 운전 상태입니다.
25	DIR	모션 방향을 표시합니다.
26	INP	인-포지션 상태입니다.
27	ORIGIN_SENSOR	원점센서가 감지 되었습니다.
28	S/W_LIMIT -	S/W – 리미트 영역에 도달 하였습니다.
29	S/W_LIMIT +	S/W + 리미트 영역에 도달 하였습니다.
30	H/W_LIMIT -	H/W – 리미트 센서가 감지 되었습니다.
31	H/W_LIMIT +	H/W + 리미트 센서가 감지 되었습니다.

Figure 10 Input Bit-Map의 비트 정보

Output Bit-Map 구조의 설명은 "5.3 IO-Map 의 구조" 항 에서 설명합니다.

■ **Object 2022h...2030h : Axis*n* Input Data**

모션게이트에 연결된 드라이브의 응답 데이터 또는 세팅제어에 대한 응답 데이터 영역입니다.

DINT	MSB			LSB
Byte	4 byte area	2 byte area	1 byte area	0 byte area
	32bit Data			

OBJECT DESCRIPTION

INDEX	2022h… 2030h
Name	Axis*n* Output Data
Object Code	VAR
Data Type	DINT

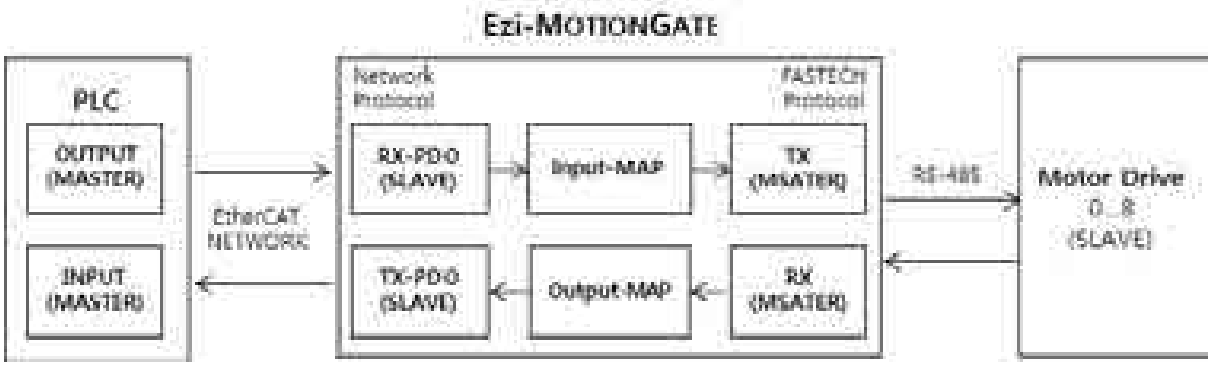
ENTRY DESCRIPTION

Access	Read Only
PDO Mapping	Yes
Value Range	DINT
Default Value	$0xFFFF_{[Hex]}$, $-1_{[dec]}$: 최초의 전원을 인가 후 Input Bit-Map 의 0 bit 가 0 일 때

5. 모션게이트

5.1 모션게이트의 개요

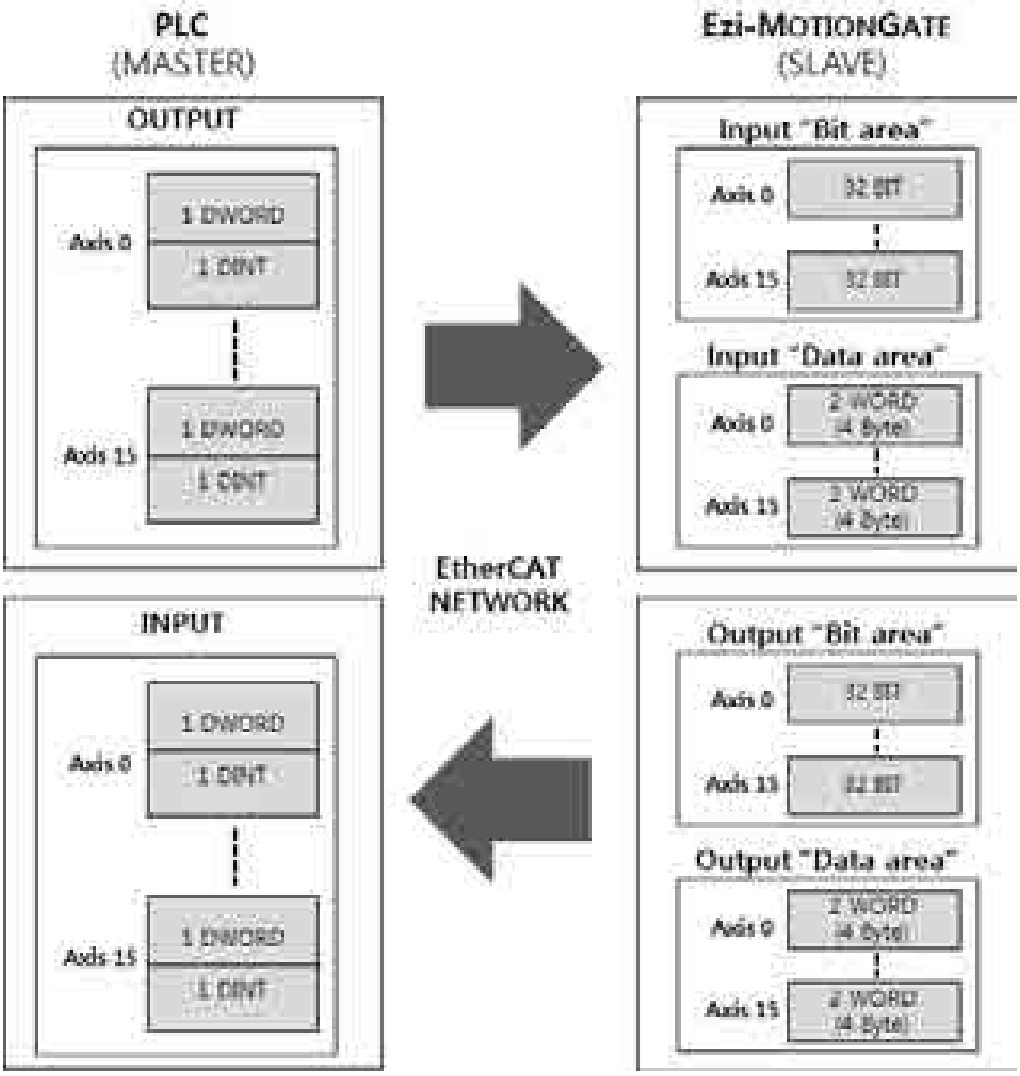
EtherCAT 네트워크에서 사용하는 모션게이트는 모션드라이브를 최대 15개(Axis 0~15)를 지원 합니다. 상위 제어기(PLC)는 모션게이트와 연결된 송수신 구간의 메모리 주소를 액세스 가능한 마스터 시스템이 되어야 합니다. 상위제어기의 리모트 데이터를 모션게이트가 EtherCAT 네트워크로부터 수신하여, 수신된 데이터의 Input-Map에 해당하는 축을 제어합니다. 각 축에서의 응답정보는 Output-Map으로 구성하여 EtherCAT 네트워크로 리모트 데이터를 송신합니다.



5.2 모션게이트의 데이터 전송 구조

EtherCAT 네트워크에서의 모션게이트는, 연결된 각 모터드라이브에 대한 명령을 IO-Map으로 구성된 데이터를 사용하여 명령 및 정보 요청 실행 합니다. 여기서 IO-Map의 어드레스 영역은 각 축에 대한 제어 명령, 응답 정보 확인을 위한 영역입니다. IO-Map의 데이터 구성은 한 축에 대하여 1 DWORD(32bit 비트 열 형식)으로 구성된 비트영역(Bit area)와 1 DINT(4 Byte 실수 형식)으로 구성된 데이터영역(Data area)으로 나뉩니다.

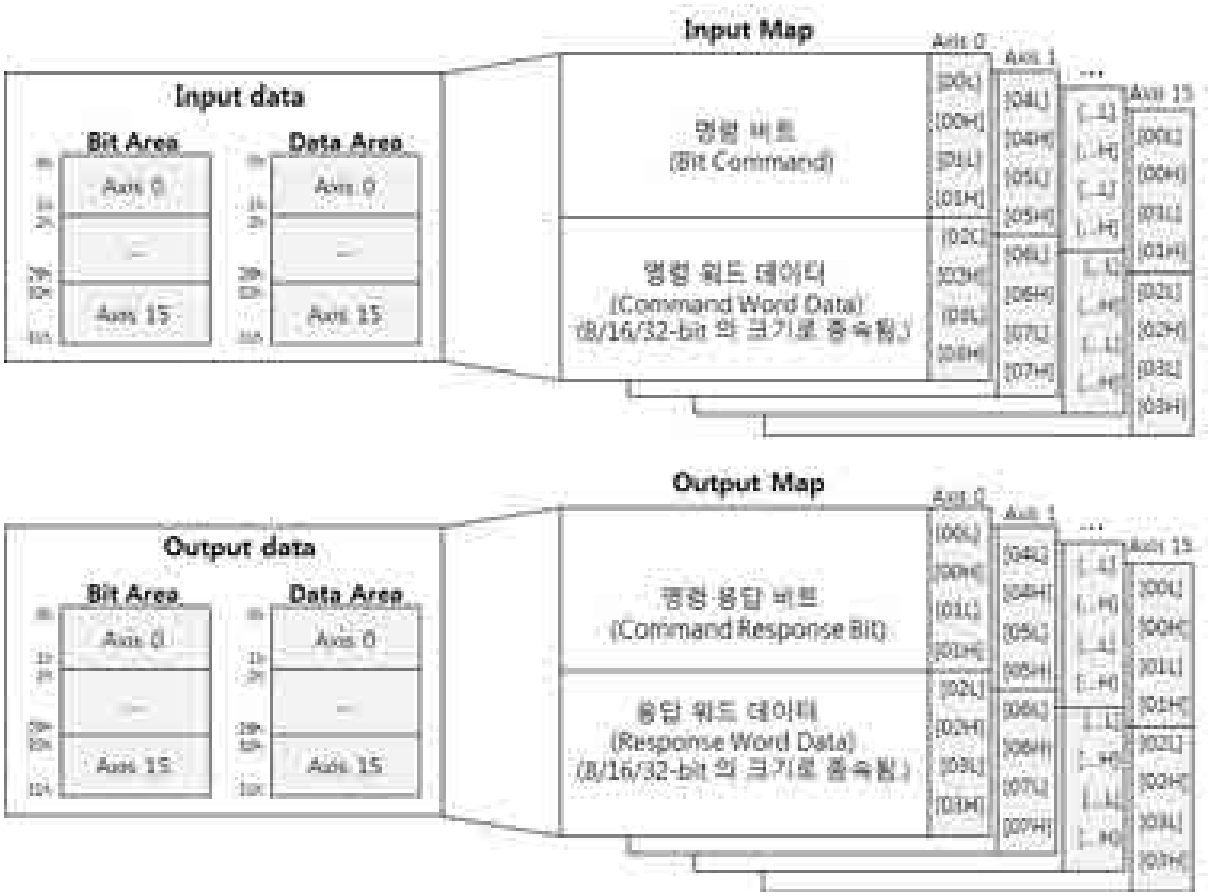
상위 제어기(PLC)는 IO-Map의 데이터 어드레스 영역으로 제어와 응답 데이터를 확인 할 수 있습니다.



5.3 IO-Map의 구조

IO-Map은 비트영역[0-3]과 데이터영역[4-7]으로 나누어 집니다. Input-Map의 비트영역은 모터축의 비트명령을 위한 구간으로 사용되며, 데이터영역은 비트명령에 해당하는 데이터 정보를 입력하는 구간으로 사용됩니다. Output-Map의 비트영역은 해당 축의 상태플래그 또는 제어 명령에 대한 명령응답 비트로 사용되고, 워드영역은 Input-Map으로 지시한 명령의 데이터가 저장되는 구간입니다.

IO-Map 의 구조는 맨 처음의 주소에서 2번째의 주소까지의 바이트 영역은 Axis 0번 대한 IO-Map이고 뒤로 Axis 1의 IO-Map으로 이어져 Axis 15의 송신구간으로 2바이트 영역으로 이어집니다.



5.3.1 명령제어 비트영역과 상태정보 비트영역

■ **Input-Map의 비트 구성**

Input-Map은 모터 드라이브의 제어를 지령 하는 영역입니다. 명령에 대한 비트의 조합으로 모터 드라이브의 모션 제어의 선택 및 응답 정보 형식 설정, 파라미터 또는 PT 정보 등의 값을 설정할 수 있습니다.

Input-Map의 비트 영역(상위 4바이트 영역)

BYTE offset	BIT	비트 이름	동작 레벨	Description
0	0	CONNECT	상승 엣지	이 비트의 설정으로 해당 축의 사용여부를 결정하며, '1'로 세트 하였을 때 해당 축과 통신연결을 시도하며, 해당 축과 통신이 필요하지 않을 경우 이 비트의 설정을 '0'으로 설정합니다. '0'으로 설정 되었을 때 모션게이트는 해당 축과의 통신처리는 제외되며, 어떠한 명령도 실행 하지 않습니다. 여러 축에 대한 명령이벤트가 동시에 발생되었을 시 모터 드라이브의 번호가 낮은 순에서 높은 순으로 처리가 됩니다. 한 축에 대한 이벤트의 처리가 완료되었다면, 다음 ID의 축에 대한 프로세스 작업으로 변경 됩니다. 해당된 축에서 명령 또는 이벤트가 없을 경우, 모션게이트는 해당 축의 상태정보와 응답요청 항목에 대한 데이터를 수신 합니다. - 해당 축의 상태정보(flags FLAG-define) - 지령 위치 값 (signed long 32-bit) - 실제 위치 값 (signed long 32-bit) - 위치 오차 값 (signed long 32-bit) - 현재 운전 속도 값 (signed long 32-bit) - 현재 운전중인 PT번호 **NOTE 1:** 모션게이트는 Fas_GetAxisStatus() 명령으로 드라이브와 통신을 합니다. **NOTE 2:** 모터제어의 지연시간은 동일한 지연시간을 갖는 모터일 경우 연결 축수에 대한 배가 됩니다.

BYTE offset	BIT	비트 이름	동작 레벨	Description
	1	ENABLE / IGNORED	상승 엣지	SERVO Drive : 해당 축의 상태를 모션동작이 가능한 상태로 전환합니다. 0 : Servo OFF 1 : Servo ON STEP Drive : 이 비트의 명령은 무시 됩니다.
	2	ESTOP	하강 엣지	모션 또는 모든 명령 실행 정지. (비상 정지) * 0: E-Stop 명령 실행, 1: E-Stop 명령 대기
	3	ALARM_RESET / MOTOR_FREE	상승 / 하강	SERVO Drive : 발생된 알람을 해제 할 때 사용 (상승 엣지 동작) * STEP Drive 는 MOTOR_FREE 비트를 '1'로 유지할 경우 STEP 드라이브는 Motor Free 상태가 유지되고, '0'으로 전환되는 하강 엣지에서 스텝모터 알람 리셋 명령이 실행 됩니다.
	4	CMD_START	상승 엣지	조그 Speed Override 명령, 위치이동, PT 운전, 원점이동의 명령을 실행 시 사용
	5	-	-	-
	6	-	-	-
	7	MOTION / SETTTING	H/L	MotionGate 의 Map 의 구성을 모션 및 설정으로 선택하는 비트 0: 모션제어 모드 1: 설정 모드
1	0	CMD_CODE0	H/L	**모션 제어 모드일 때** 0000(0): 일반 모션(Jog, Step, 영점 이동) 0001(1): 상대치 이동[Incrometal Move], 절대치 이동[Absolute Move] 0100(4): PT 운전 (PT 운전, 싱글 PT 운전) 0111(7): 원점 이동 (Origin) **설정 모드 일 때** 0000(0): 명령 없음 0101(5): 버전 정보 확인 1000(8): 파라미터 요청 1001(9): 파라미터 쓰기 1010(10): 위치정보 변경 1100(12): 알람내역 요청 1101(13): 알람내역 삭제 1110(14): 파라미터 저장
	1	CMD_CODE1		
	2	CMD_CODE2		
	3	CMD_CODE3		

BYTE offset	BIT	비트 이름	동작 레벨	Description
	4	RESPONSE_TYPE0	H/L	해당 축에 대한 RX 구간으로부터 수신을 원하는 응답데이터의 응답형식을 지정 0000(0): 응답데이터를 요청하지 않음. 0001(1): 지령 위치 0010(2): 실제 위치 0011(3): 위치 오차 0100(4): 현재 속도 0101(5): 운전중인 PT 번호 1000(8): 현재 발생된 알람 번호 ***설정 모드에서는 사용되지 않습니다**.
	5	RESPONSE_TYPE1		
	6	RESPONSE_TYPE2		
	7	RESPONSE_TYPE3		
2	0	CANCEL	상승 엣지	모션에 대한 일반 정지
	1	HOLD	상승 엣지	모션 중 일시 정지
	2	-	-	-
	3	GO_ZERO_POS	상승 엣지	해당 축의 드라이브에서 지정된 영점으로 이동 (위치 값: 0)
	4	-JOG_MOV	상승 엣지	역방향 조그(JOG)운전 데이터 영역의 입력 값: 속도 비율, 속도 값, Speed Step 번호.
	5	+JOG_MOV	상승 엣지	정 방향 조그(JOG)운전 데이터 영역의 입력 값: 속도 비율, 속도 값, Speed Step 번호.
	6	-STEP_MOV	상승 엣지	모션게이트의 내부 파라미터 값인 위치 값과, 속도 값으로 가감이동 데이터 영역의 입력 값: 위치 값의 번호(0~3) *사용자에 의하여 재 정의 가능
	7	+STEP_MOV	상승 엣지	모션게이트의 내부 파라미터 값인 위치 값과, 속도 값으로 증감이동 워드영역의 입력 값: 위치 값의 번호(0~3) *사용자에 의하여 재 정의 가능

BYTE offset	BIT	비트 이름	동작 레벨	Description
3	0	INC/ABS	H/L	제어 방법이 위치이동일 때(CMD_CODE:0001) 상대치 이동 또는 절대치 이동을 선택하는 비트 0: 상대치 이동 1: 절대치 이동
	1	-	-	-
	2	SPD_MODE	H/L	제어 방법이 일반 모션일 때(CMD_CODE: 0000) Jog 이동 시 사용 0: 입력된 비율 값 또는 Speed Step 번호로 Jog 운전 1: 입력된 속도 값으로 Jog 운전
	3	-	-	-
	4	SINGLE_PT	H/L	제어 방법이 PT 운전일 때(CMD_CODE:0100), 일반 PT 운전, 또는 싱글 PT 운전을 선택하는 비트. 0: 일반 PT 운전 1: 싱글 PT 운전
	5	-	-	-
	6	-	-	-
	7	-	-	-

NOTE: Input Map은 모션게이트가 PLC 또는 Master로 명령을 입력 받는 영역 입니다.

■ **Output-Map의 비트 구성**

Output-Map의 구간은 데이터 플래그와 비트명령에 대한 루프-백(Loop-Back: 되돌림)비트가 존재합니다. 루프-백 비트는 해당 비트의 명령 이벤트와 동일하게 반응하는 비트로 Input-Map의 비트 입력 여부를 확인할 수 있습니다. 상태 플래그는 해당 모터드라이브와 통신하여 수신된 데이터 정보를 기반으로 나타납니다.

Output-Map의 비트 영역(상위 4바이트 영역)

Byte offset	bit	비트 이름	동작 레벨	비트 유형	Description
0	0	CONNECTED	H	상태 비트	해당 축의 Plus-R 과 연결되었을 경우 이 비트는 '1'로 세트
	1	ENABLED MOTOR_FREE_RESP (STEP)	H	상태 비트	해당축의 Servo ON 또는 스텝모터의 노말 상태일 때 '1'로 세트 *STEP Drive 일 때, Motor Free 명령에 대한 응답 비트가 됨.
	2	ESTOP_RESP	H	루프-백	Input-Map 의 nESTOP 비트의 루프-백 비트로 비상정지 명령이 실행되면 '1'로 세트
	3	ALARM_ERROR	H	상태 비트	해당 축의 모터 드라이브에서 알람이 발생되었을 때 자동으로 '1'로 세트, 알람이 해제되면 '0'으로 클리어
	4	CMD_RESP	H	루프-백	Input-Map 의 CMD_START 비트의 루프-백 비트
	5	OUT_RANGE	H	상태 비트	Input-Map 의 데이터영역의 값이 해당 명령의 값의 범위에 맞지 않았을 때 '1'로 세트
	6	READY	H	상태 비트	현재 해당 축에 대한 명령이 실행 가능한 상태일 때 '1'로 세트 됩니다. 이 비트가 '0'인 상태에서 어느 명령도 동작하지 않습니다. **NOTE 1:** 세팅 모드에서 READY 비트가 '1'로 세트 되었을 때, 다른 축의 제어가 가능합니다.
	7	SET_MOV_RESP	H/L	루프-백	현재 Output-Map 의 데이터가 세팅 모드일 때는 '1'로 세트 되고, 모션모드일 때 '0'으로 클리어

Byte offset	bit	비트 이름	동작 레벨	비트 유형	Description
	0	CMD_CODE_RESP0	H/L	루프-백	명령 받은 모션의 종류를 응답 0000(0): 일반 모션(Jog, Step, 영점 이동) 0001(1): 상대이동[Incrometal Move], 절대값이동[Absolute Move] 0100(2): PT 운전 (PT 운전, 싱글 PT 운전) 0111(3): 원점 이동 (Origin)
	1	CMD_CODE_RESP 1	H/L		
	2	CMD_CODE_RESP 2	H/L		
	3	CMD_CODE_RESP 3	H/L		
	4	RESPONSE_TYPE_RESP0	H/L	루프-백	Word 영역에 지정되는 되는 응답데이터의 정보를 응답 0000(0): 응답데이터를 요청하지 않음. 0001(1): 명령위치 0010(2): 실제 위치 0011(3): 위치 오차 0100(4): 현재 속도 0101(5): 운전중인 PT 번호 1000(8): 현재 발생된 알람 번호
	5	RESPONSE_TYPE_RESP1	H/L		
	6	RESPONSE_TYPE_RESP2	H/L		
	7	RESPONSE_TYPE_RESP3	H/L		
2	0	MOTIONNING	H/L	상태 비트	해당 축의 모션상태일 때 '1'로 세트
	1	HOLD_RESP	H/L	상태 비트	운전 중 HOLD 비트의 명령으로 일시 정지된 상태일 때 '1'로 세트
	2	-			
	3	GO_ORIGIN_RESP	H	상태 비트	해당 축의 Plus-R 의 파라미터로 원점복귀를 실행 중 일 때, '1'로 세트
	4	-	-		-
	5	JOG_RESP	H	상태 비트	해당 축이 Jog 운전 중일 때
	6	-	-		-
	7	STEP_RESP	H	상태 비트	해당 축이 Step 운전 중일 때

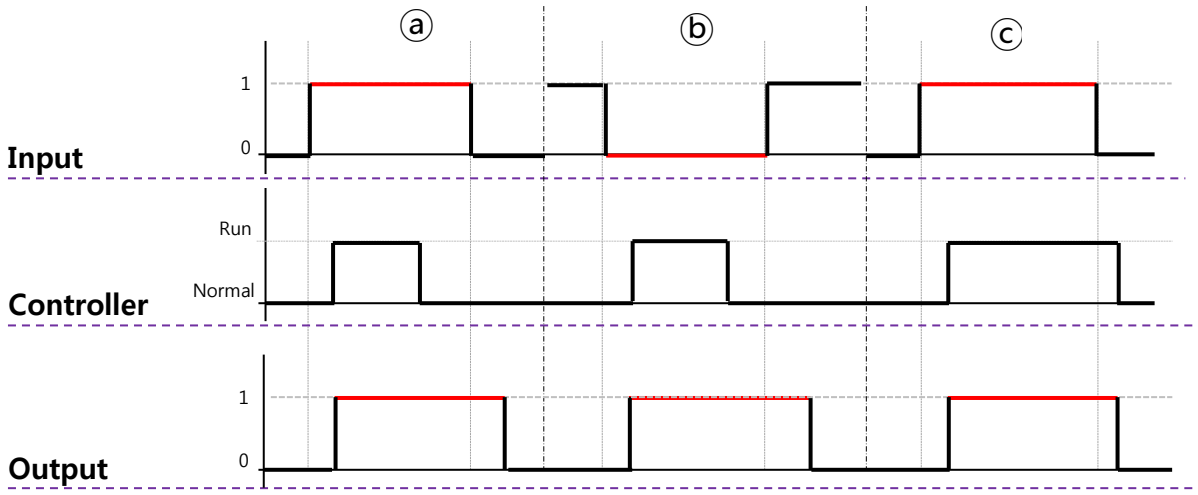
Byte offset	bit	비트 이름	동작 레벨	비트 유형	Description
	0	PT_RUNNING	L/H	상태 비트	해당 축이 위치이동 중일 때
	1	MOV DIR	L/H	상태 비트	모터의 회전방향을 표시 0 : CW(+) 1 : CCW(-) * 이 비트를 확인 시 FLAG_IN_MOTION 비트가 '1'로 세트 되었을 때 갱신된 값으로 확인해야 합니다. 논리연산 (FLAG_IN_MOTION & FLAG_nDIR)
	2	INP	L/H	상태 비트	모터가 In position 이 완료되었을 때 '1'로 세트 됩니다. * 모터가 STEP 일 경우 이 비트는 동작되지 않습니다.
	3	ORIGIN_SENSOR	H	상태 비트	원점센서가 ON 이 된 경우 '1'로 세트 됩니다.
	4	SW_LIMIT_N	H	상태 비트	'-' 방향 프로그램 리미트를 초과한 경우 '1'로 세트 됩니다.
	5	SW_LIMIT_P	H	상태 비트	'+' 방향 프로그램 리미트를 초과한 경우 '1'로 세트 됩니다.
	6	HW_LIMIT_N	H	상태 비트	'-' 방향 리미트 센서가 ON 이 된 경우 '1'로 세트 됩니다.
	7	HW_LIMIT_P	H	상태 비트	'+' 방향 리미트 센서가 ON 이 된 경우 '1'로 세트 됩니다.

NOTE: Output Map은 모션게이트가 PLC 또는 Master로 상태정보를 출력하는 영역 입니다.

5.3.2 IO-Map의 동작 순서 및 동작 조건

■ IO-Map의 명령 비트 실행 방법

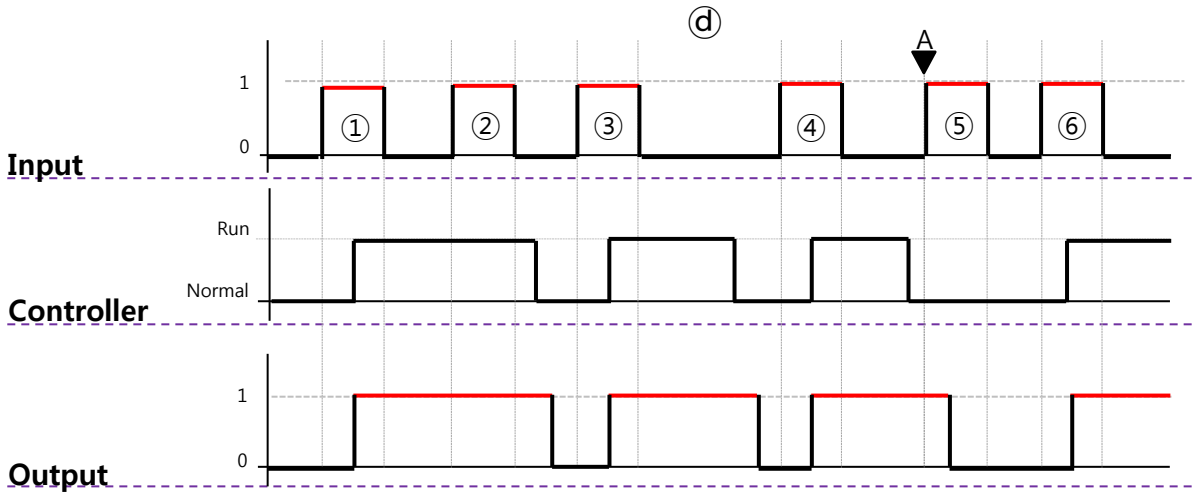
비트 명령은 상승엣지와 하강엣지 명령으로 구분됩니다.



Input의 상승 엣지의 명령 시작은 ⓐ구간과 같이 '0' 상태에서 '1'로 변경되는 시점입니다. 이 명령을 받은 모션게이트는 해당 축으로 명령을 하달하고, 그 명령이 실행될 때 Output으로 명령에 대하여 응답합니다.

하강 엣지의 명령 시작은 는 ⓑ의 구간은 와 같이 Input 명령이 '1' 상태에서 '0'으로 변경되는 시점입니다. 이 이벤트로 모션게이트는 해당 축으로 명령을 하달하고, 그 명령이 실행 될 때 Output으로 명령에 대하여 응답합니다.

ⓒ구간과 같은 비트 명령은 Input의 상승엣지 명령으로 모션게이트가 해당 축의 동작명령을 하달하여, 하강 엣지의 명령이 있을 때까지 지속적인 명령이 유지 되는 명령 입니다. 이 명령의 순서는 Input 의 상승 엣지로 해당축의 동작이 되면, 동작에 대하여 Output으로 응답 합니다. 그리고 Input의 하강 엣지 명령으로 해당축의 동작이 정지 되면, Output으로 동작의 정지 대하여 응답됩니다.



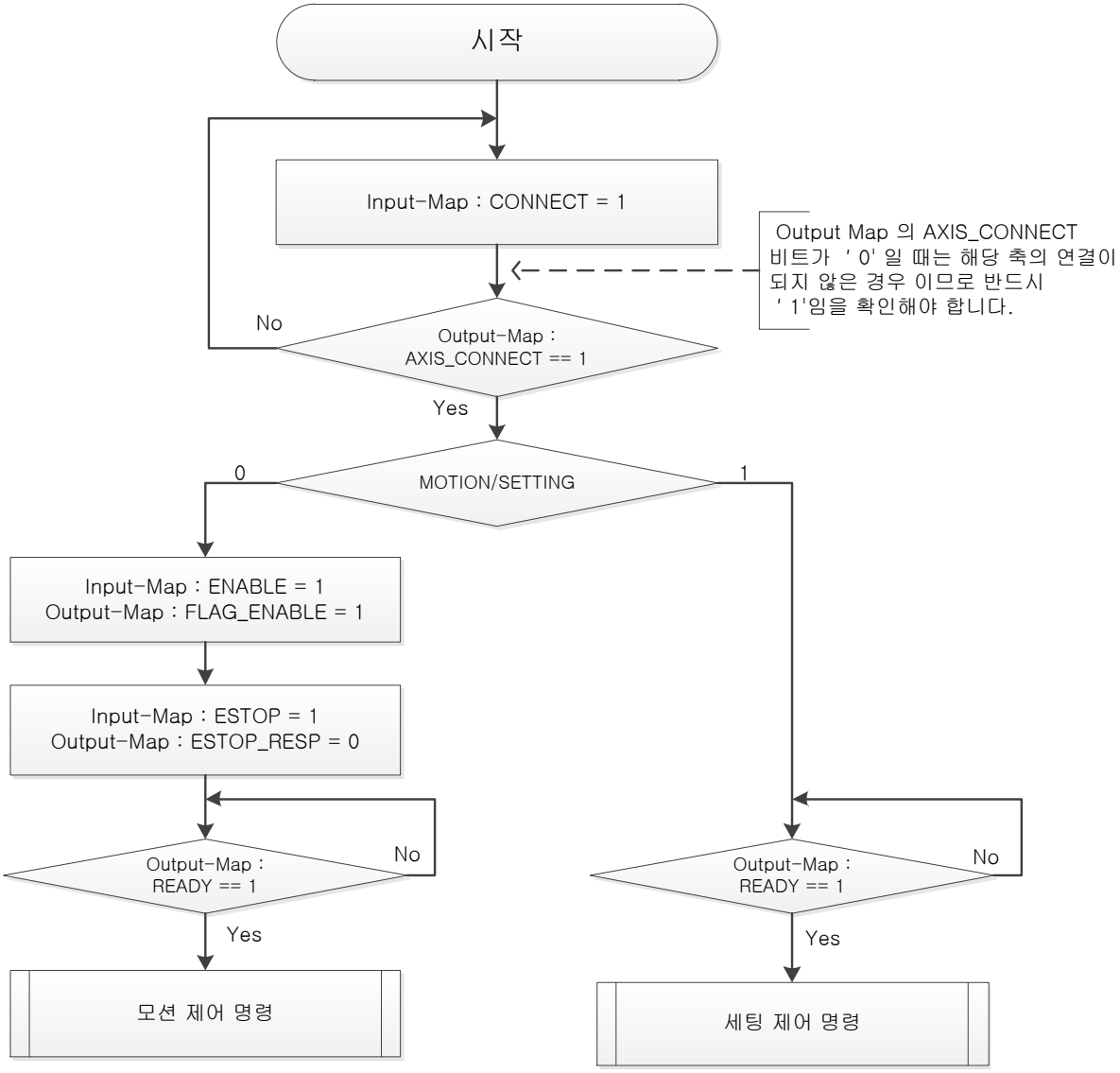
ⓓ구간은 Input 명령이 연속적으로 동작 했을 경우 입니다. 이러한 경우에는 ①의 명령으로 ⓐ 구간과 동일하게 명령이 시작됩니다. 이때 모션게이트가 동작 중일 때 입력된 명령 ② 으로 동작되지 않습니다. 그리고 ①의 명령으로 동작한 모션게이트의 동작이 완료 된 후 입력된 ③의 명령으로는 동작이 실행됩니다.

④의 명령으로 실행된 동작이 완료되고 Output 의 응답하기 전의 시점인 A에서 ⑤의 명령이 입력되면, 이 입력은 무시됩니다. 그러나 Output이 응답된 후에 입력된 ⑥의 명령으로 동작합니다. 즉, Input 명령으로 모션게이트의 동작이 실행되고, Output에 동작이 완료된 상태의 응답이 있을 때의 Input명령이 유효 합니다.

■ **IO-Map의 제어 명령 준비 순서**

모션게이트는 명령을 실행할 때 아래의 순서의 과정이 필요 합니다.

순서도 1. 모션 및 설정 제어 명령의 활성화 조건



※ 모션게이트의 명령

① Input-Map 의 CONNECT 비트를 '1'로 세트 하여 명령을 실행 <참고: IO-Map 사용 설명서 *2.2.1>

- CONNECT 비트는 해당축의 사용을 선택하는 비트 이므로 반드시 '1'로 세트
- Output-Map 의 AXIS_CONNECT 비트의 응답 상태가 '1'임을 확인

② Input-Map 의 MOTION/SETTING 비트를 선택 <참고 : IO-Map 사용 설명서 *2.1>

- 모션 제어는 '0', 세팅 제어는 '1'로 선택 합니다.

③ 모션 제어는 Input-Map 의 ENABLE 비트와 ESTOP 비트를 '1'로 세트 <참고: IO-Map 사용 설명서 *2.2.1>

- Output-Map 의 응답비트 FLAG_ENABLE 비트 '1'임을 확인
- ESTOP_RESP 비트가 '0' 을 확인

④ 명령을 실행 하고자 할 때 Output-Map 의 READY 비트의 상태를 확인 <참고: IO-Map 사용 설명서 * 2.7>

- 다른 명령이 실행 중 일 때 READY 비트는 '0' 상태로 유지 됨.
- 모션 명령이 없을 때 READY 비트는 '1' 상태로 유지 됨.
- 세팅 명령을 할 때, 해당 명령이 완료 될 때까지 '0' 상태로 유지 됨.

⑤ 드라이브의 모션 제어는 모션 명령에 대한 IO-Map 의 비트 조합으로 수행합니다.

<참고: IO-Map 사용 설명서 *2.1.1, *2.1.2>

- Input-Map 의 MOTION/SETTING 비트를 '0'으로 설정
- 모션 제어의 명령 실행은 "CONNECT= 1, ENABLE=1, nESTOP=1" 을 반드시 설정 해야 함

⑥ 드라이브 및 모션게이트의 설정 값에 대한 확인 및 수정은 세팅 제어에 대한 IO-Map 의 비트 조합으로 수행 됨 <참고: IO-Map 사용 설명서 *6. *7. *8. *9>

- Input-Map 의 MOTION/SETTING 비트를 '1'로 설정
- 세팅 제어의 명령 실행은 "CONNECT= 1" 을 반드시 설정 해야 함

6. IO-Map의 사용 예제

본 예제는 오므론 PLC의 NJ 시리즈로 제작된 예제입니다. ST 명령 기반의 FB(Function block) 제작되었으며, FB에는 필요에 따라서 중복된 명령이 포함 되어있습니다.

예제에서 설명하는 FB는 모든 축에 대한 명령을 지원하도록 제작되었으며, In/Out 부분에 명령을 실행할 드라이브의 IO-Map 변수를 적용하여 사용할 수 있습니다.

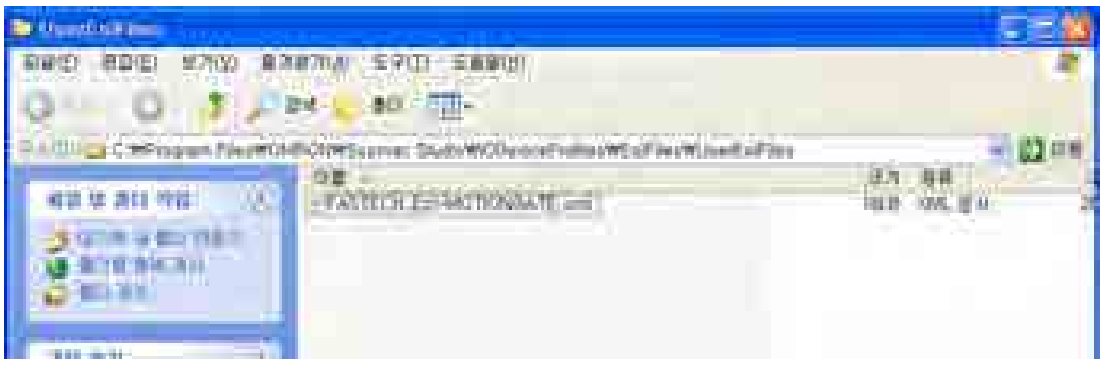
FB은 ST 언어로 제작되었으며, 본 예제에서 기재된 ST 예문은 ST 언어가 지원되는 다른 EtherCAT 마스터에서 사용 가능합니다.(이때, 사용되는 마스터의 프로그램에서 FB를 제작하십시오)

6.1 PLC 프로그램 설정

6.1.1 ESI 파일 등록

ESI(EtherCAT Slave Information)파일은 EtherCAT 슬레이브장치의 정보를 EtherCAT 마스터에 등록하는 파일로 모션게이트의 ESI파일은 "FASTECH_Ezi-MOTIONGATE.xml" 로 제공합니다.

① ESI 파일을 EtherCAT 마스터의 설정 프로그램의 ESI 파일 등록절차에 준하여 다음과 같이 포함합니다.



② ESI 파일이 정상 등록 되었는지 확인 합니다.

③ ESI 파일이 정상적으로 등록이 되었을 경우, 다음 그림과 같이 제품정보가 표시 됩니다.

④ 우측 도구 상자에 제품 그룹에 "FASTECH Motion Gateways"를 선택하게 되면, 모션게이트의 아이콘을 확인할 수 있습니다.

6.1.2 EtherCAT ID 및 변수 등록

■ **EtherCAT ID 등록**

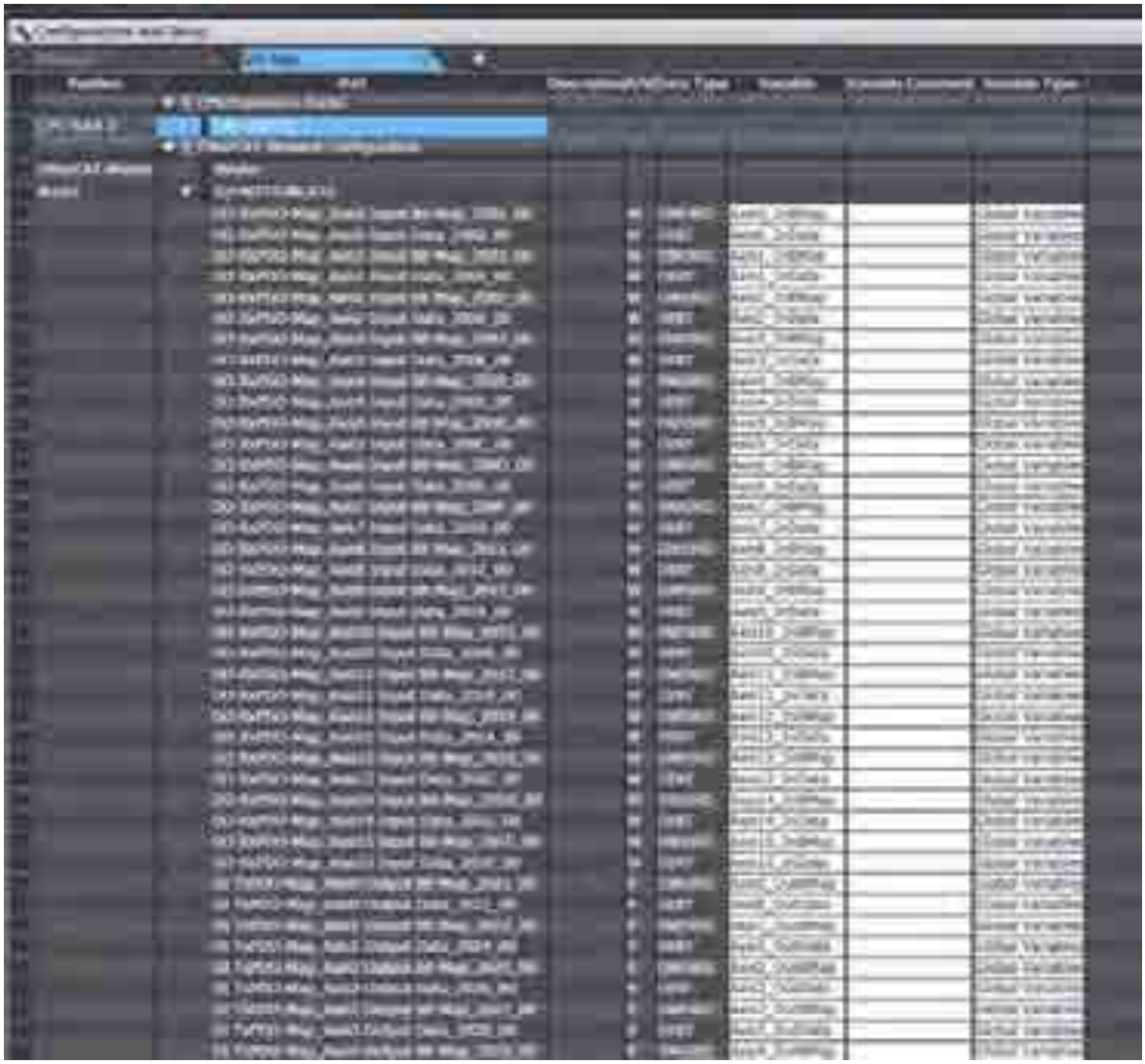
EtherCAT ID는 모션게이트의 EtherCAT Device ID 설정 스위치(SW1~SW3)로 설정 가능합니다. 여기서 ID를 1000 이상의 값으로 설정 하기 위해서는 스위치 설정 값을 "0 0 0"으로 설정 후, 마스터에서 EtherCAT Slave ID를 지정하여 사용 가능합니다. 이 방법은 EtherCAT 마스터에서 모션게이트의 ESC(EtherCAT Slave Controller)에 접근하여 EtherCAT ID를 변경 하는 방법입니다. 이 작업을 완료 후 모션게이트를 반드시 리-부트 해야 정상적으로 ID가 설정 됩니다.

⚠주의	마스터를 사용한 ID 변경은 EtherCAT 마스터의 제조사의 방법에 따르십시오. EtherCAT Device ID 를 변경 후 반드시 모션게이트를 리-부트 하십시오.

■ **마스터에서의 EtherCAT I/O Mapping**

① 좌측 Configuration and Setup 창에서 I/O Map 창을 선택합니다.

② 이 창을 띄우게 되면 모션게이트에 대한 POD 정보를 확인 할 수 있습니다.

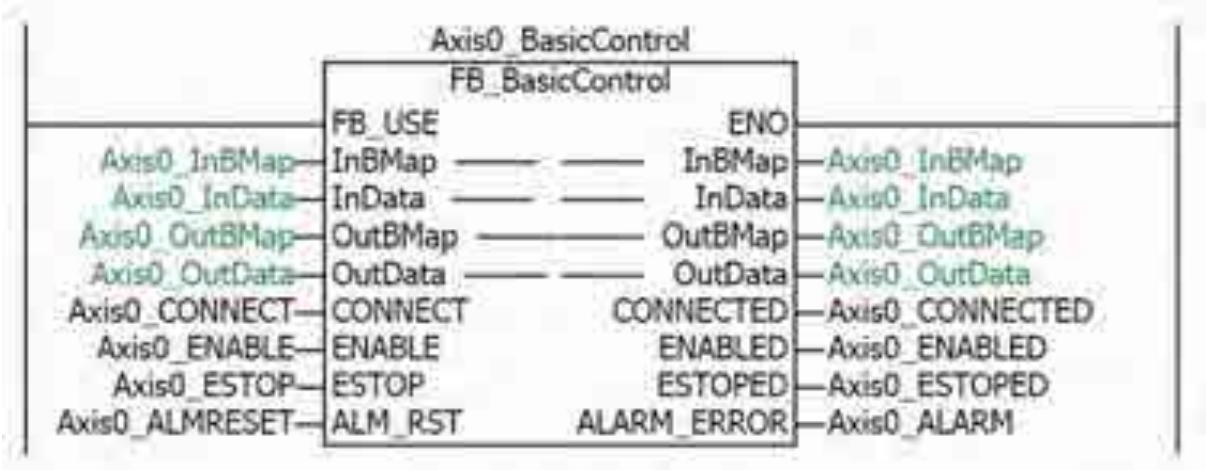


③ POD 정보는 "4.5.2 Manufacturer Specific Objects" 의 내용을 참조 하십시오.

6.2 기본 제어

기본제어 명령은 모션게이트와 드라이브간의 통신 연결 명령과, 드라이브의 Servo ON/OFF 제어, 비상정지 명령, 알람 리셋 명령으로 구분 됩니다. (다음 FB는 PLC I/O맵 할당에서 모션게이트의 Axis-0 에 대한 전역변수를 사용한 예제 입니다.)

Function Block 1. 기본제어 명령



FB_BasicControl의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DINT	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DINT	No Edge		FALSE	FALSE
CONNECT	Input	BOOL	No Edge		FALSE	FALSE
CONNECTED	Output	BOOL	No Edge		FALSE	FALSE
ENABLE	Input	BOOL	No Edge		FALSE	FALSE
ENABLED	Output	BOOL	No Edge		FALSE	FALSE
ESTOP	Input	BOOL	No Edge		FALSE	FALSE
ESTOPED	Output	BOOL	No Edge		FALSE	FALSE
ALARM_ERROR	Output	BOOL	No Edge		FALSE	FALSE
ALM_RST	Input	BOOL	No Edge		FALSE	FALSE

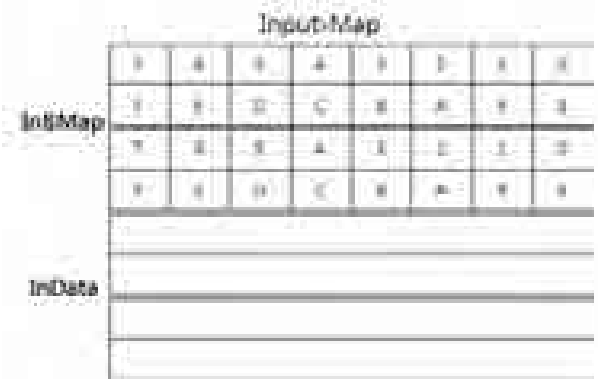
6.2.1 CONNECT

CONNECT 명령은 해당축의 사용 결정시 사용합니다.

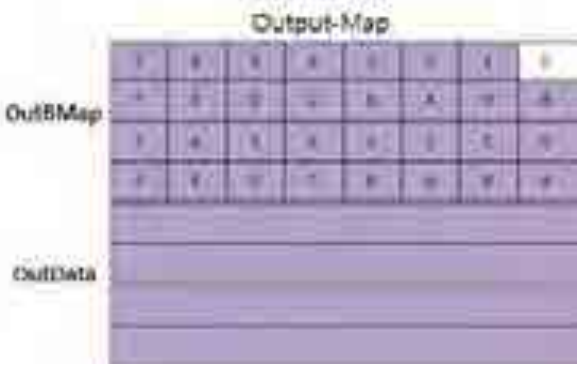
ST 1. CONNECT 명령 처리

```
//Connect Command
IF CONNECT THEN
      InBMap:= InBMap OR DWORD#16#00000001;
ELSE
      InBMap:= InBMap AND DWORD#16#FFFFFFFE;
END_IF;

//Connect Response bit check
IF (OutBMap AND DWORD#16#00000001) = 16#00000001
THEN
     CONNECTED:= TRUE;
ELSE
     CONNECTED:= FALSE;
END_IF;
```

- **CONNECT 명령의 실행 전**

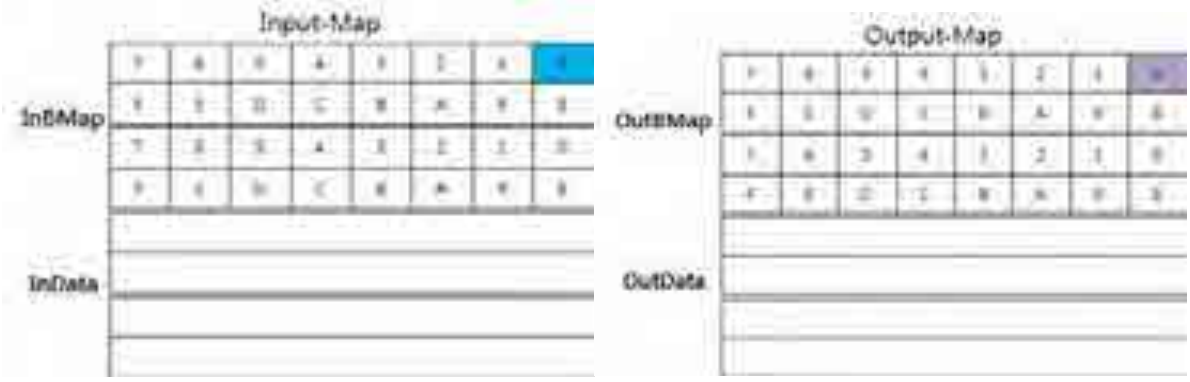


CONNECT bit = 0



CONNECTED bit = 0

- **CONNECT 명령의 실행 후**



CONNECT bit = 1 CONNECTED bit = 1

6.2.2 ENABLE 명령과 E-STOP 명령

E-STOP명령은 ESTOP 비트가 비활성화 된 상태(ESTOP bit = 0일 때)에서 동작됩니다. 따라서, ENABLE 명령을 실행하기 위해서는 E-STOP명령을 '1'로 설정하여 실행합니다.

ST 2. ENABLE 명령 처리

```
//Enable Command
IF ENABLE THEN
        InBMap:= InBMap OR DWORD#16#00000002;
ELSE
        InBMap:= InBMap AND DWORD#16#FFFFFFD;
END_IF;
// Enable Status Response bit check
IF (OutBMap AND DWORD#16#00000002) = 16#00000002 THEN
        ENABLED:= TRUE;
ELSE
        ENABLED:= FALSE;
END_IF;
```

ST 3. E-STOP 명령 처리

```
//Emergency Stop Inverse Command
IF ESTOP THEN
        InBMap:= InBMap OR DWORD#16#00000004;
ELSE
        InBMap:= InBMap AND DWORD#16#FFFFFFB;
END_IF;

// Emergency Stop Response bit check
IF (OutBMap AND DWORD#16#00000004) = 16#00000004 THEN
        ESTOPED:= TRUE;
ELSE
        ESTOPED:= FALSE;
END_IF;
```

- **ENABLE 명령의 실행**

ENABLE bit = 1
E-STOP bit = 1

ENABLED bit = 1
E-STOP RESP bit = 0

- **E-STOP 명령을 실행**

ENABLE bit = 1
E-STOP bit = 0

ENABLED bit = 0
E-STOP RESP bit = 1

6.2.3 ALARM 상태 확인 및 ALARM 리셋

알람상태는 ALM/ERR 비트로 확인할 수 있습니다. 알람 확인 후 알람 해제는 ALARM_RESET 비트의 활성화(ALARM_RESET = 1)로 실행됩니다.

ST 4. ALARM 확인 및 ALARM Reset

```
//ALARM Reset Command
IF ALM_RST THEN
        InBMap:= InBMap OR DWORD#16#00000008;
ELSE
        InBMap:= InBMap AND DWORD#16#FFFFFF7;
END_IF;

// ALARM Status bit check
IF (OutBMap AND DWORD#16#00000008) = 16#00000008
THEN
        ALARM_ERROR:= TRUE;
ELSE
        ALARM_ERROR:= FALSE;
END_IF;
```

- **알람 발생 시**

ALARM RESET bit = 0
ENABLE bit = 1
E-STOP bit = 1

ALARM bit = 1
ENABLED bit = 0
E-STOP RESP bit = 0

- **알람 해제 명령을 실행**

ALARM RESET bit = 1

ENABLE bit = 1

E-STOP bit = 1

ALARM bit = 0

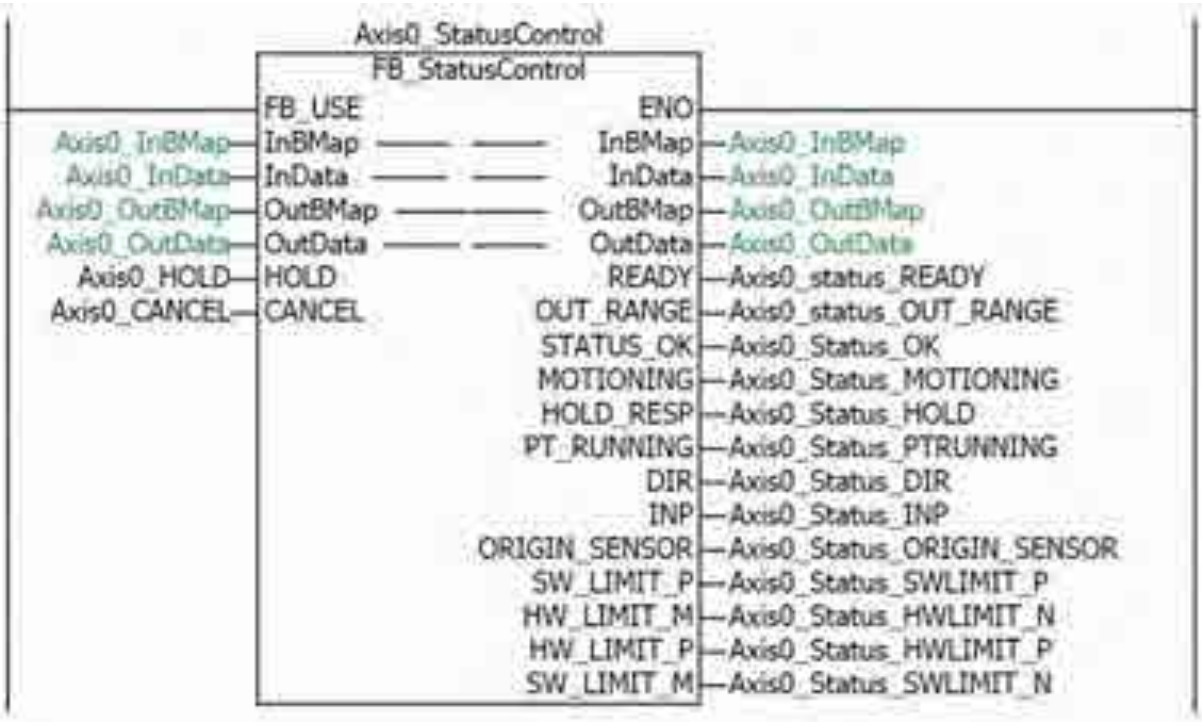
ENABLED bit = 0

E-STOP RESP bit = 0

6.3 상태 제어

상태 제어는 드라이브의 운전 정지 및 상태 정보 확인을 실행 압니다. 이 제어는 모션모드(MOTIONION)상태에서 가능 합니다.

Function Block 2. 상태 제어 명령



FB_StatusControl의 사용 변수 목록

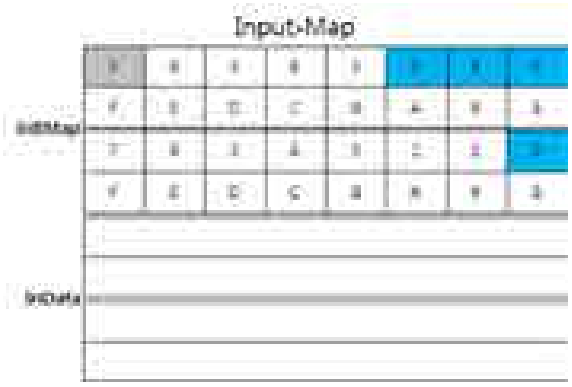
Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DINT	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DINT	No Edge		FALSE	FALSE
CANCEL	Input	BOOL	No Edge		FALSE	FALSE
HOLD	Input	BOOL	No Edge		FALSE	FALSE
READY	Output	BOOL	No Edge		FALSE	FALSE

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
OUT_RANGE	Output	BOOL	No Edge		FALSE	FALSE
STATUS_OK	Output	BOOL	No Edge		FALSE	FALSE
MOTIONING	Output	BOOL	No Edge		FALSE	FALSE
HOLD_RESP	Output	BOOL	No Edge		FALSE	FALSE
PT_RUNNING	Output	BOOL	No Edge		FALSE	FALSE
DIR	Output	BOOL	No Edge		FALSE	FALSE
INP	Output	BOOL	No Edge		FALSE	FALSE
ORIGIN_SENSOR	Output	BOOL	No Edge		FALSE	FALSE
SW_LIMIT_P	Output	BOOL	No Edge		FALSE	FALSE
HW_LIMIT_M	Output	BOOL	No Edge		FALSE	FALSE
HW_LIMIT_P	Output	BOOL	No Edge		FALSE	FALSE
SW_LIMIT_M	Output	BOOL	No Edge		FALSE	FALSE

6.3.1 CANCLE

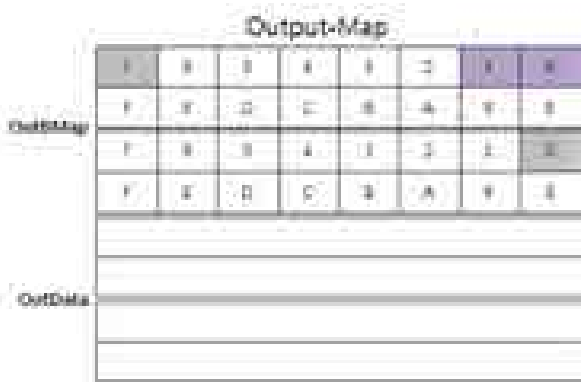
실행취소 명령은 모션정지, 일시정지 명령 취소, PT운전 정지를 위하여 사용됩니다.

✓ **IO-Map 의 명령 및 응답 형태**



MOTION/SETTING bit = 0

CANCLE bit = 1



MOTION/SETTING_RESP bit = 0

MOTIONING bit = 0

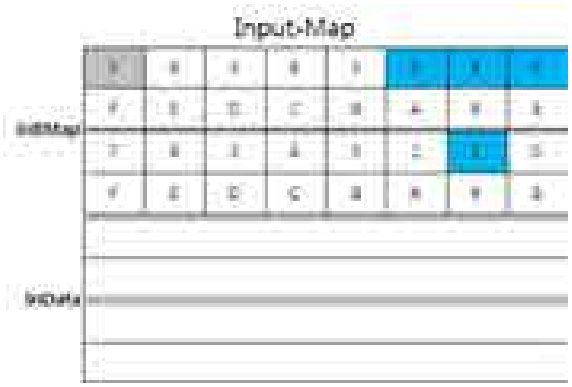
ST 5. CANCLE 명령 처리

```
//CANCEL Command
IF CANCEL THEN
        InBMap:= InBMap AND DWORD#16#FFFFFF7F;      //MOTION/SETTING bit Clear
        InBMap:= InBMap OR DWORD#16#00010000;       //CANCEL Bit SET
ELSE
        InBMap:= InBMap AND DWORD#16#FFFEFFFF;      //CANCEL Bit Clear
END_IF;
```

6.3.2 HOLD

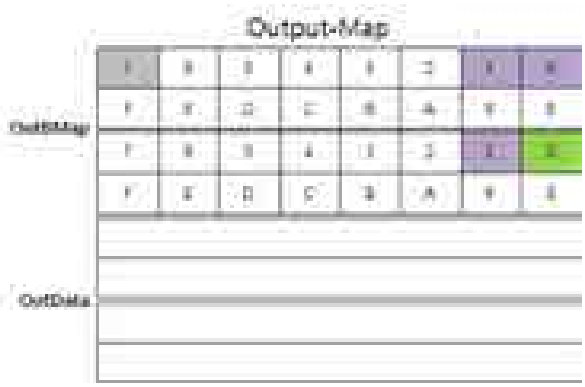
일시정지 명령은 모션 명령을 일시적으로 정지하고, 재개 할 수 있도록 하는 명령입니다. 일시정지 명령으로 모션이 정지되면, HOLD_RESP 비트는 '1'로 세트 되고, MOTIONING 비트는 '0'으로 클리어 됩니다. 모션 재개 명령을 실행 하였을 때는 HOLD_RESP 비트는 '0'로 클리어 되고, MOTIONING 비트는 '1'으로 세트 됩니다. 만약 모션이 실행되지 않은 상태에서 일시 정지 명령과, 모션 재개 명령을 실행 하였을 경우에는 MOTIONING 비트는 '0'상태로 유지 됩니다.

✓ **IO-Map 의 명령 및 응답 형태**



MOTION/SETTING bit = 0

HOLD bit = 1



MOTION/SETTING_RESP bit = 0

HOLD_RESP bit = 1

MOTIONING bit = 0

ST 6. HOLD 명령 처리

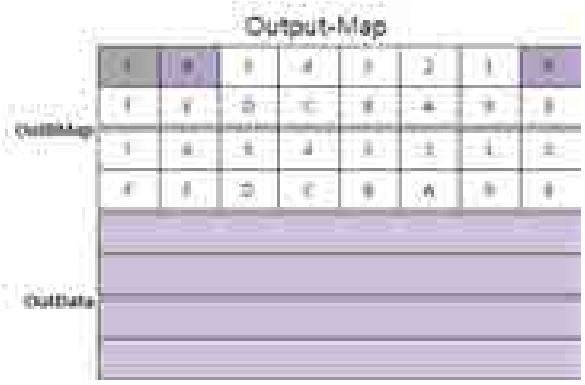
```
//HOLD Command
IF HOLD THEN
        InBMap:= InBMap AND DWORD#16#FFFFFF7F;      //MOTION/SETTING bit Clear
        InBMap:= InBMap OR DWORD#16#00020000;       //HOLD Bit SET
ELSE
        InBMap:= InBMap AND DWORD#16#FFFDFFFF;      //HOLD Bit Clear
END_IF;
```

6.3.3 상태 정보 확인

상태 정보 확인은 모션모드(MOTIONION)상태에서 확인할 수 있습니다.

- **READY**

IO-Map에 대한 명령이 가능 할 때 ON됩니다.

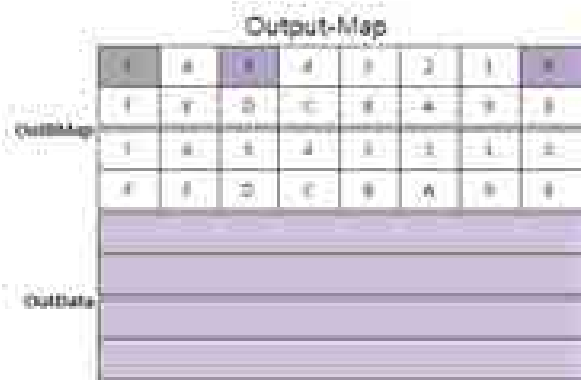


ST 7. READY 비트 처리

```
//READY Bit
IF (OutBMap AND DWORD#16#00000040)=16#00000040      THEN
        READY := TRUE;
ELSE
        READY := FALSE;
END_IF;
```

- **OUT_RANGE**

IO-Map에 대한 명령시 데이터 범위를 초과 하였을 때 ON 됩니다.



ST 8. OUT_RANGE 비트 처리

```
//OUT Range Bit
IF (OutBMap AND DWORD#16#00000020)=16#00000020      THEN
        OUT_RANGE := TRUE;
ELSE
        OUT_RANGE := FALSE;
END_IF;
```

※ IO-Map의 모드 설정 (Motion Mode, Setting Mode)

Motion Mode 의 IO-Map

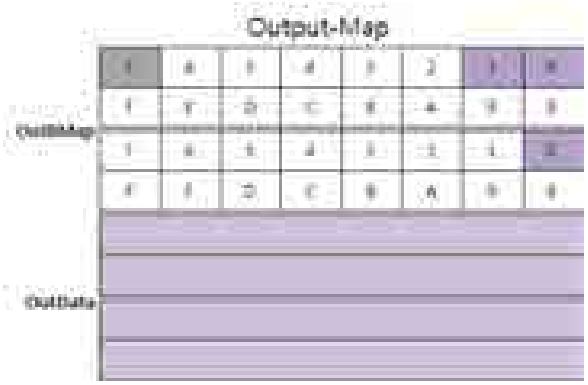
Setting Mode 의 IO-Map

ST 9. IO-Map 모드 설정

```
//Mode Set
IF (OutBMap AND DWORD#16#00000080) = 16#00000080    THEN     // Setting Mode
        Motion_Mode := FALSE;
        STATUS_OK:=FALSE;
ELSE      // Motion Mode
        Motion_Mode := TRUE;
        STATUS_OK:=TRUE;
END_IF;
```

■ **MOTIONING**

모터가 구동중일 때 ON 됩니다.

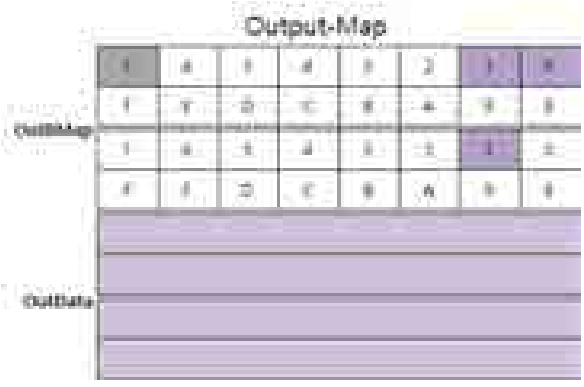


ST 10. MOTIONING 비트 처리

```
IF Motion_Mode THEN
        IF (OutBMap AND DWORD#16#00010000)=16#00010000      THEN
                MOTIONING := TRUE;
        ELSE      MOTIONING := FALSE;
END_IF;
```

■ **HOLD**

운전중 일시정지 상태로 되었을 때 ON 됩니다.

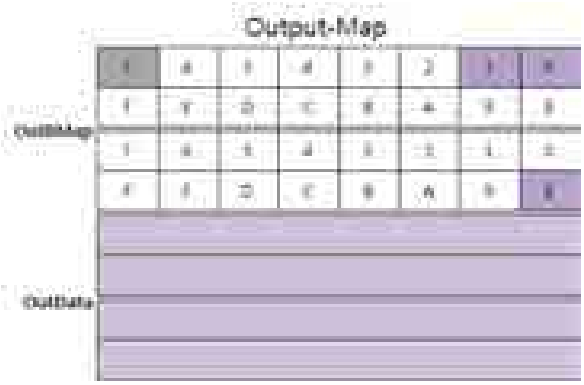


ST 11. HOLD 응답 비트 처리

```
IF (OutBMap AND DWORD#16#00020000)=16#00020000      THEN
        HOLD_RESP := TRUE;
ELSE
        HOLD_RESP := FALSE;
END IF;
```

■ **PTRUNNING**

PT운전 상태일 때 ON 됩니다.

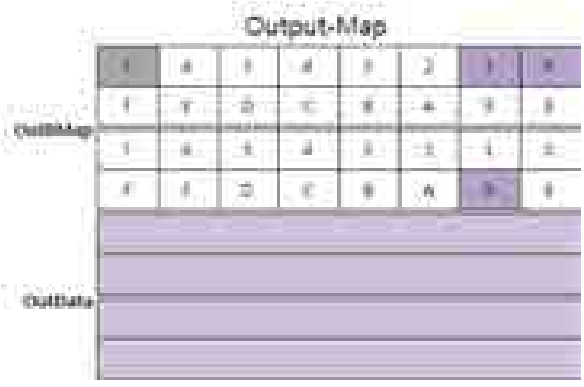


ST 12. MOTIONING 비트 처리

```
IF (OutBMap AND DWORD#16#01000000)=16#01000000      THEN
        PT_RUNNING := TRUE;
ELSE
        PT_RUNNING := FALSE;
END_IF;
```

■ **DIR**

모터의 회전방향이 정방향(CW)상태일 때 ON 됩니다.

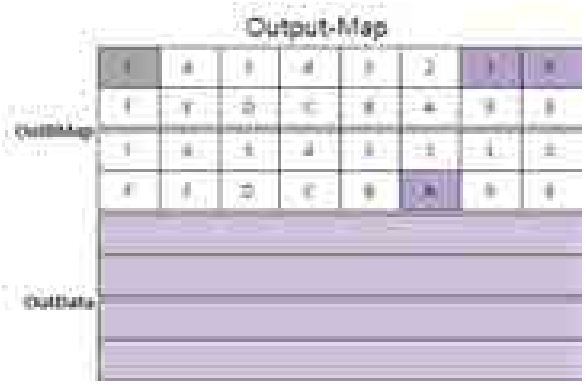


ST 13. DIR 비트 처리

```
IF (OutBMap AND DWORD#16#02000000)=16#02000000 THEN
        DIR := TRUE;
ELSE
        DIR := FALSE;
END IF;
```

■ **INP**

Inposition 이 완료된 상태일 때 ON 됩니다.

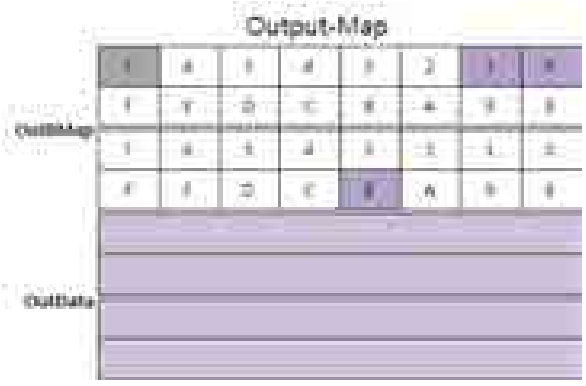


ST 14. INP 비트 처리

```
IF (OutBMap AND DWORD#16#04000000)=16#04000000      THEN
        INP := TRUE;
ELSE
        INP := FALSE;
END IF;
```

■ **ORIGIN_SENSOR**

원점센서가 ON 되어 있는 상태일 때 ON 됩니다.

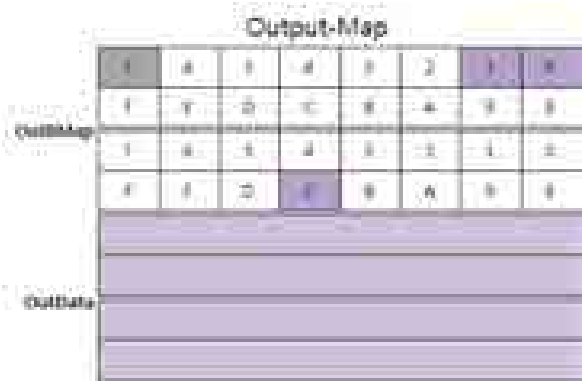


ST 15. ORIGIN 비트 처리

```
IF (OutBMap AND DWORD#16#08000000)=16#08000000      THEN
        ORIGIN_SENSOR := TRUE;
ELSE
        ORIGIN_SENSOR := FALSE;
END IF:
```

■ **SW_LIMIT -**

-방향 프로그램 리미트를 초과한 경우일 때 ON 됩니다.

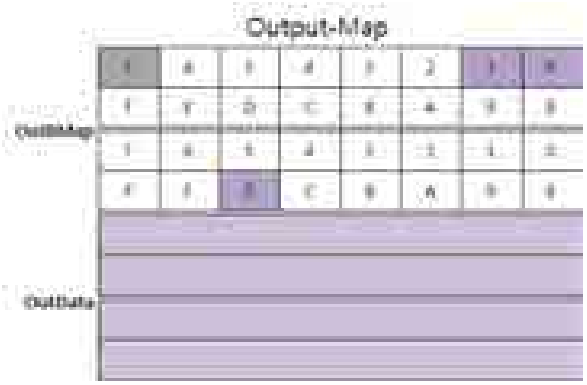


ST 16. SW_LIMIT- 비트 처리

```
IF (OutBMap AND DWORD#16#10000000)=16#10000000      THEN
        SW_LIMIT_M := TRUE;
ELSE
        SW_LIMIT_M := FALSE;
END IF;
```

■ **SW_LIMIT +**

+방향 프로그램 리미트를 초과한 경우일 때 ON 됩니다.

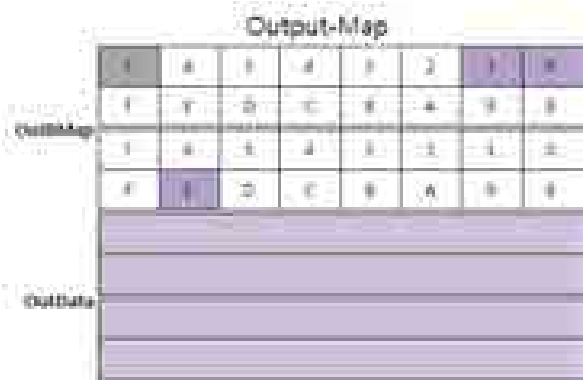


ST 17. SW_LIMIT+ 비트 처리

```
IF (OutBMap AND DWORD#16#20000000)=16#20000000      THEN
        SW_LIMIT_P := TRUE;
ELSE
        SW_LIMIT_P := FALSE;
END_IF;
```

■ **HW_LIMIT -**

-방향 리미트 센서가 ON된 경우일 때 ON 됩니다.

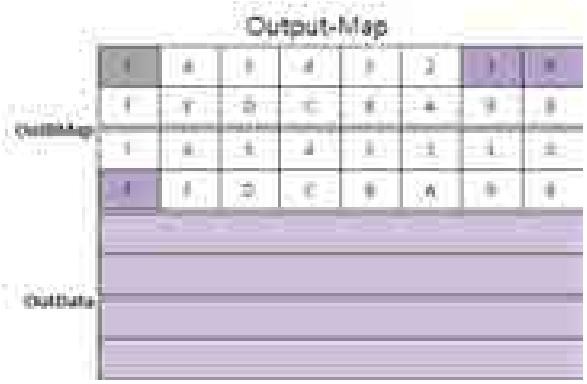


ST 18. HW_LIMIT- 비트 처리

```
IF (OutBMap AND DWORD#16#40000000)=16#40000000      THEN
          HW_LIMIT_M := TRUE;
ELSE
          HW_LIMIT_M := FALSE;
END_IF;
```

■ **HW_LIMIT +**

+방향 리미트 센서가 ON 된 경우일 때 ON 됩니다.



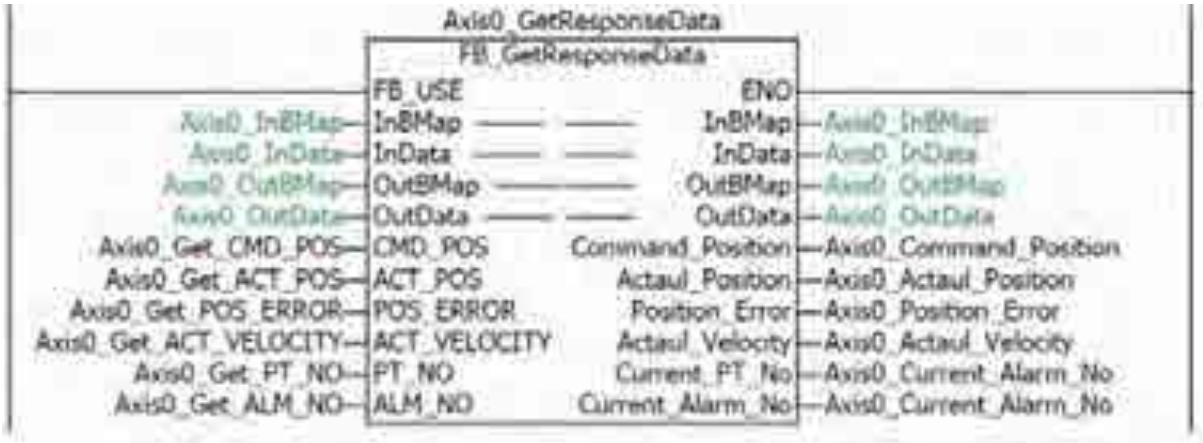
ST 19. HW_LIMIT- 비트 처리

```
IF (OutBMap AND DWORD#16#80000000)=16#80000000      THEN
          HW_LIMIT_P := TRUE;
ELSE
          HW_LIMIT_P := FALSE;
END IF;
```

6.4 응답 데이터 설정

모션게이트의 응답 데이터(RESPONSE TYPE)는 드라이브의 목표 위치, 현재 위치, 위치 오차, 현재 운전속도, PT 번호, 알람 발생시 해당 알람의 번호를 얻을 수 있습니다. 이 설정은 IO-Map 이 모션 모드일 때, 확인 됩니다.

Function Block 3. 응답 데이터 확인



FB_GetResponseData의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DINT	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DINT	No Edge		FALSE	FALSE
CMD_POS	Input	BOOL	No Edge		FALSE	FALSE
ACT_POS	Input	BOOL	No Edge		FALSE	FALSE
POS_ERROR	Input	BOOL	No Edge		FALSE	FALSE
ACT_VELOCITY	Input	BOOL	No Edge		FALSE	FALSE
PT_NO	Input	BOOL	No Edge		FALSE	FALSE
ALM_NO	Input	BOOL	No Edge		FALSE	FALSE
Command_Position	Output	DINT	No Edge		FALSE	FALSE

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
Actual_Position	Output	DINT	No Edge		FALSE	FALSE
Position_Error	Output	DINT	No Edge		FALSE	FALSE
Actual_Velocity	Output	DINT	No Edge		FALSE	FALSE
Current_PT_No	Output	DINT	No Edge		FALSE	FALSE
Current_Alarm_No	Output	DINT	No Edge		FALSE	FALSE
Response_Type	Internals	RESP_TYPE			FALSE	FALSE

데이터 타입- Enumerated

Type Name	Member Name	Enum Value	Comment
RESP_TYPE			열거 유형
	RESP_NO_RESP	0	응답데이터 없음
	RESP_CMD_POS	1	위치추종 데이터
	RESP_ACT_POS	2	현재 위치 데이터
	RESP_POS_ERROR	4	위치 오차
	RESP_ACT_VELOCITY	5	구동 속도
	RESP_ALARM_NO	8	알람 정보

ST 20. 응답 데이터 설정 및 확인 명령 처리

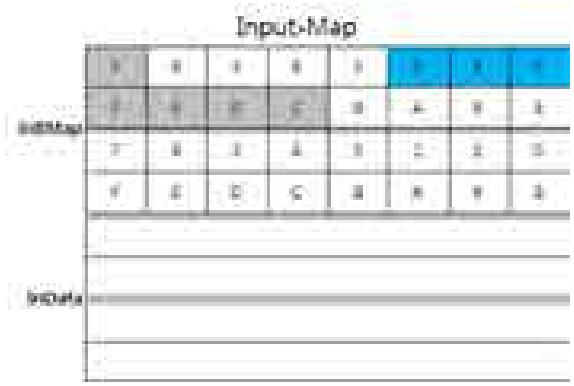
```
//Mode Setting
IF (OutBMap AND DWORD#16#00000080) = 16#00000080 THEN
        // Setting Mode
        Motion_Mode := FALSE;
ELSE
        // Motion Mode
        Motion_Mode := TRUE;
END_IF;

IF CMD_POS THEN
    Response_Type :=RESP_CMD_POS;
```

```
END_IF;
IF ACT_POS THEN
    Response_Type := RESP_ACT_POS;
END_IF;
IF POS_ERROR THEN
    Response_Type := RESP_POS_ERROR;
END_IF;
IF ACT_VELOCITY THEN
    Response_Type := RESP_ACT_VELOCITY;
END_IF;
IF PT_NO THEN
    Response_Type := RESP_PT_NO;
END_IF;
IF ALM_NO THEN
    Response_Type := RESP_ALARM_NO;
END_IF;

// Only Motion Mode
IF Motion_Mode THEN
    //Response Data type Set
    CASE Response_Type OF
        RESP_NO_RESP :
            InBMap:= (InBMap AND DWORD#16#FFFF0FFF) OR DWORD#16#00000000;
        RESP_CMD_POS :
            InBMap:= (InBMap AND DWORD#16#FFFF0FFF) OR DWORD#16#00001000;
        RESP_ACT_POS :
            InBMap:= (InBMap AND DWORD#16#FFFF0FFF) OR DWORD#16#00002000;
        RESP_POS_ERROR :
            InBMap:= (InBMap AND DWORD#16#FFFF0FFF) OR DWORD#16#00003000;
        RESP_ACT_VELOCITY :
            InBMap:= (InBMap AND DWORD#16#FFFF0FFF) OR DWORD#16#00004000;
        RESP_PT_NO :
            InBMap:= (InBMap AND DWORD#16#FFFF0FFF) OR DWORD#16#00005000;
        RESP_ALARM_NO :
            InBMap:= (InBMap AND DWORD#16#FFFF0FFF) OR DWORD#16#00008000;
    ELSE
        InBMap:= InBMap AND DWORD#16#FFFF0FFF;
    END_CASE;
```

```
    //Get Response Data
    CASE DWORD_TO_DINT(OutBMap AND DWORD#16#0000F000) OF
        16#00001000 : Command_Position := OutData;
        16#00002000 : Actual_Position := OutData;
        16#00003000 : Position_Error := OutData;
        16#00004000 : Actual_Velocity := OutData;
        16#00005000 : Current_PT_No:= OutData;
        16#00008000 : Current_Alarm_No:= OutData;
    END_CASE;
ELSE
    ENO := FALSE;    // Not Motion mode
END_IF;
```

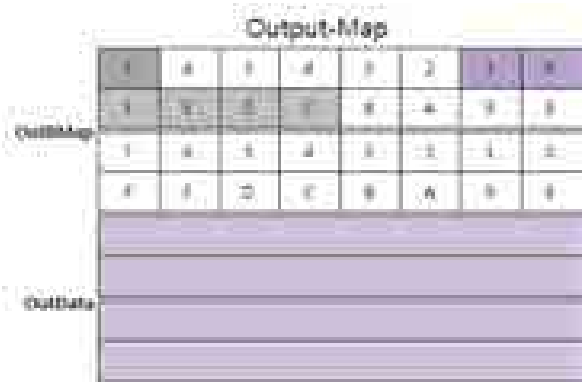
■ **No Response Data**

응답 데이터를 요청하지 않습니다.



MOTION/SETTING bit = 0

RESPONSE_TYPE 0~3 = 0000b



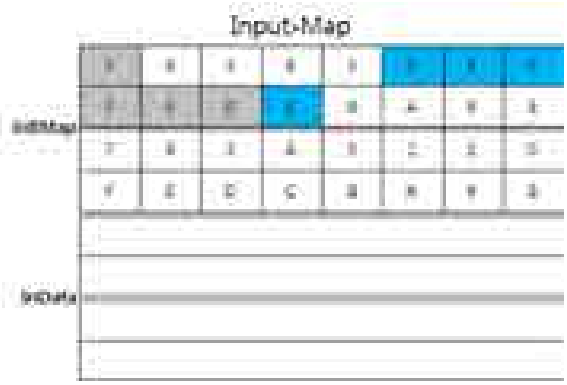
MOTION/SETTING_RESP bit = 0

RESPONSE_TYPE 0~3 = 0000b

RESPONSE_DATA = 0

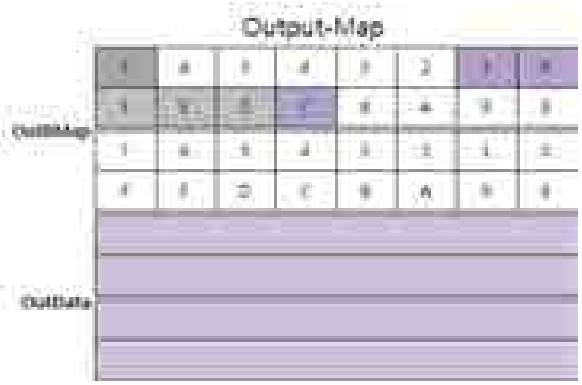
■ **Command Position**

위치 지령 값을 요청합니다.



MOTION/SETTING bit = 0

RESPONSE_TYPE 0~3 = 0001b



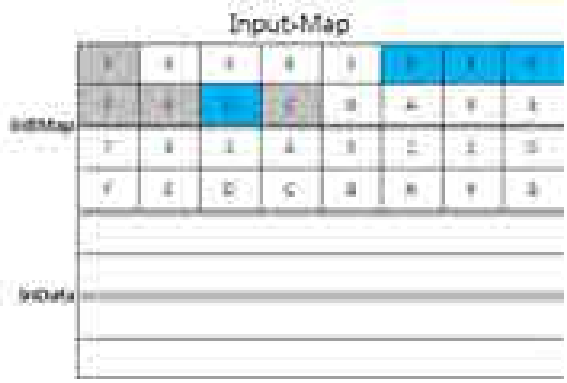
MOTION/SETTING_RESP bit = 0

RESPONSE_TYPE 0~3 = 0001b

RESPONSE_DATA = Command Position

■ **Actual Position**

현재 위치값을 요청 합니다.



MOTION/SETTING bit = 0

RESPONSE_TYPE 0~3 = 0010b

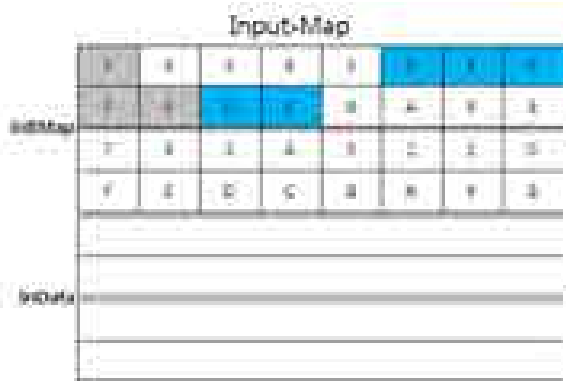
MOTION/SETTING_RESP bit = 0

RESPONSE_TYPE 0~3 = 0010b

RESPONSE_DATA = Actual Position

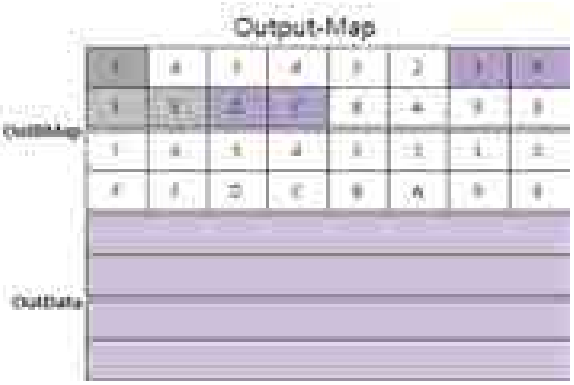
■ **Position Error**

위치 오차값(Command Position – Actual Position)을 요청 합니다.



MOTION/SETTING bit = 0

RESPONSE_TYPE 0~3 = 0011b



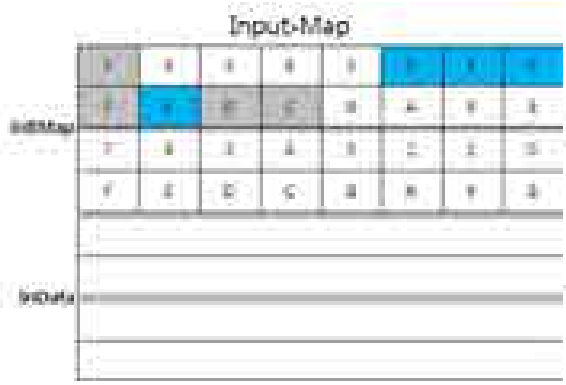
MOTION/SETTING_RESP bit = 0

RESPONSE_TYPE 0~3 = 0011b

RESPONSE_DATA = Position Error

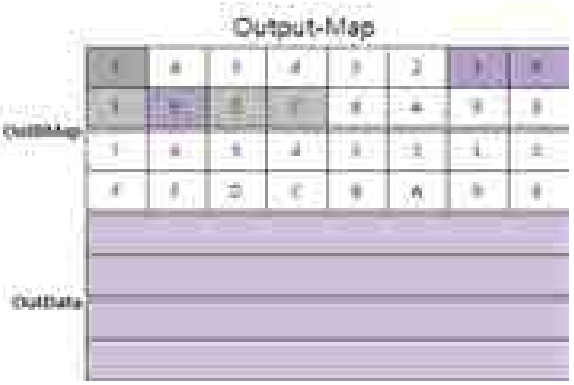
■ **Actual Velocity**

IO-Map에 대한 명령이 가능 할 때 ON됩니다.



MOTION/SETTING bit = 0

RESPONSE_TYPE 0~3 = 0101b



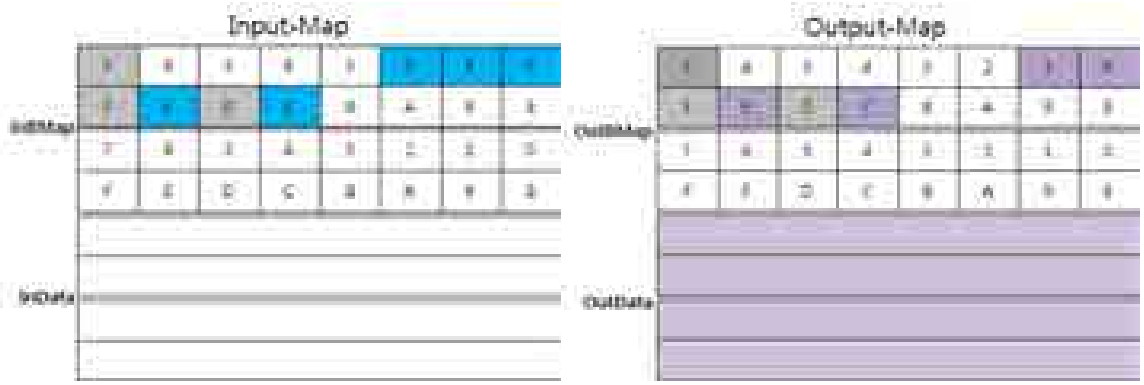
MOTION/SETTING_RESP bit = 0

RESPONSE_TYPE 0~3 = 0101b

RESPONSE_DATA = Actual Velocity

■ **Current PT No**

IO-Map에 대한 명령이 가능 할 때 ON됩니다.



MOTION/SETTING bit = 0

RESPONSE_TYPE 0~3 = 0101b

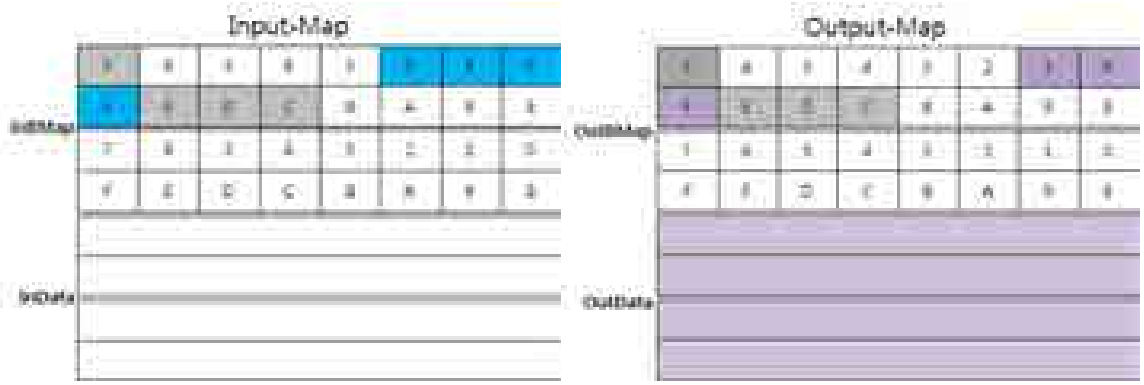
MOTION/SETTING_RESP bit = 0

RESPONSE_TYPE 0~3 = 0101b

RESPONSE_DATA = Current PT No.

■ **Current Alarm No**

IO-Map에 대한 명령이 가능 할 때 ON됩니다.



MOTION/SETTING bit = 0

RESPONSE_TYPE 0~3 = 0100b

MOTION/SETTING_RESP bit = 0

RESPONSE_TYPE 0~3 = 0100b

RESPONSE_DATA = Current Alarm Info.

6.5 모션 제어

모션제어에는 IO-Map 을 모션모드로 하여 조그 운전, 스텝 이동, 영점이동, 위치이동, PT 운전, 원점이동 명령이 있습니다.

6.5.1 조그 운전

조그 운전은 모션모드(MOTIONION)상태의 명령 코드(CMD_CODE) '0'에서 동작합니다. Speed Step Move 또는 Speed Ratio Move 과 Speed Value 의 조그 운전 명령 응답 확인은 JOG_Resp. 비트로 확인 가능합니다.

Function Block 4. JOG 운전

FB_Motion_Jog_Run의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DINT	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DINT	No Edge		FALSE	FALSE
JOGN_Move	Input	BOOL	No Edge		FALSE	FALSE

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
JOGP_Move	Input	BOOL	No Edge		FALSE	FALSE
JOG_SpdStepNo	Input	DINT	No Edge		FALSE	FALSE
JOGN_SpdMove	Input	BOOL	No Edge		FALSE	FALSE
JOGP_SpdMove	Input	BOOL	No Edge		FALSE	FALSE
JOG_SpdValue	Input	DINT	No Edge		FALSE	FALSE
JOG_SpdOverride	Input	BOOL	No Edge		FALSE	FALSE
JOG_SpdOverrideValue	Input	DINT	No Edge		FALSE	FALSE
JogRun_Resp	Output	BOOL	No Edge		FALSE	FALSE
OUT_RANGE	Output	BOOL	No Edge		FALSE	FALSE
cmd_check	Internals	BOOL				

ST 21. Jog Run 명령 처리

```
// Speed Step/Speed Ratio Run Mode   Command
IF JOGN_Move THEN
      InData:=JOG_SpdStepNo;
      InBMap:=(InBMap AND DWORD#16#0000F01F) OR DWORD#16#00100000;
ELSIF JOGP_Move THEN
      InData:=JOG_SpdStepNo;
      InBMap:=(InBMap AND DWORD#16#0000F01F) OR DWORD#16#00200000;

// Velocity Run Mode   Command
ELSIF JOGN_SpdMove THEN
      InData:=JOG_SpdValue;
      InBMap:=(InBMap AND DWORD#16#0000F01F) OR DWORD#16#04100000;
ELSIF JOGP_SpdMove THEN
      InData:=JOG_SpdValue;
      InBMap:=(InBMap AND DWORD#16#0000F01F) OR DWORD#16#04200000;
ELSE
      InBMap:=InBMap AND DWORD#16#FBCFFFFF; // JOG bit Clear
END_IF;
```

```
// Speed Override Command
IF (JOGN_Move OR JOGP_Move OR JOGN_SpdMove OR JOGP_SpdMove) THEN
      IF JOG_SpdOverride THEN
            InData:=JOG_SpdOverrideValue;
            InBMap:=InBMap OR DWORD#16#00000010;
            cmd_check:=TRUE;   //cmd_check set
      ELSE
            IF cmd_check THEN
                cmd_check := FALSE;
                InBMap:=InBMap AND DWORD#16#FFFFFFEF;   //CMD_START bit Clear
            END_IF;
      END_IF;
END_IF;


//OUT Range Bit
IF (OutBMap AND DWORD#16#00000020)=16#00000020 THEN
      OUT_RANGE := TRUE;
ELSE
      OUT_RANGE := FALSE;
END_IF;

//JOG RUN Command Response Check
IF (OutBMap AND DWORD#16#00200001) = 16#00200001 THEN
      JogRun_Resp:=TRUE;
ELSE
      JogRun_Resp:=FALSE;
END_IF;
```

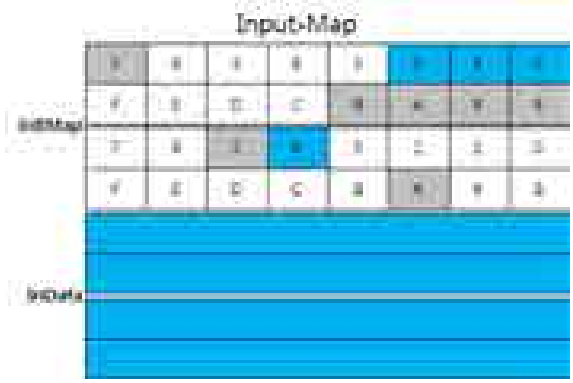
■ JOG Move – Speed Step Move or Speed Ratio Move

조그 운전의 Speed Step Move 과 Speed Ratio Move 는 운전 명령 방법은 동일합니다. 이 운전 방법은 모션게이트 파라미터 PN#0104『Use Jog Speed Ratio』의 설정 값으로 선택 됩니다.

조그 운전의 Speed Step Move 는 저장된 0~3 의 속도 단계로 조그 운전하는 모션입니다. 조그 운전의 Speed Ratio Move 는 저장된 파라미터의 PN#0105『 Move Speed for Jog Move: Ratio』의 비율로 운전합니다.

✓ **IO-Map 의 명령 및 응답 형태**

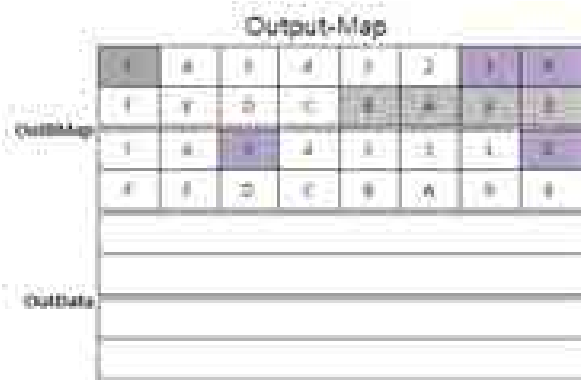
- **-JOG 명령 실행**



MOTION/SETTING bit = 0

CMD_CODE = 0000b , SPD_MODE bit = 0;

+JOG bit = 0, -JOG bit = 1

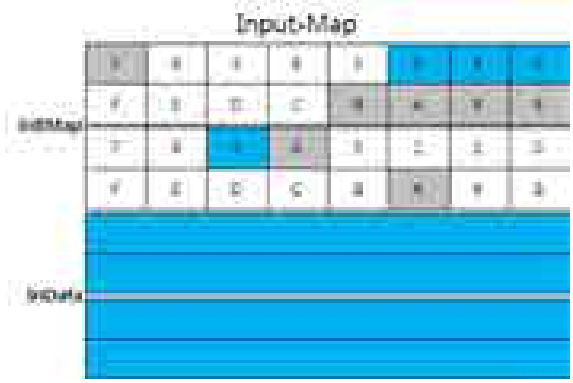


MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0000b

JOG_RESP bit = 1, MOTIONING bit = 1

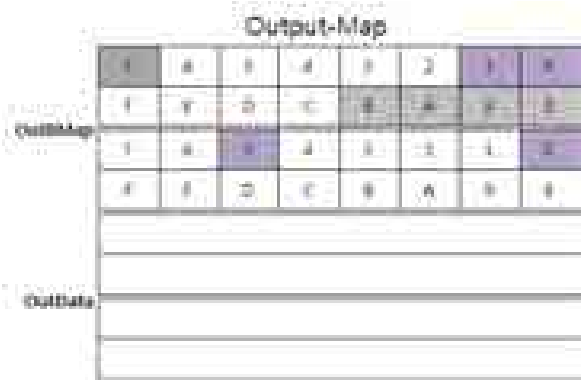
- **+JOG 명령 실행**



MOTION/SETTING bit = 0

CMD_CODE = 0000b , SPD_MODE bit = 0;

+JOG bit = 1, -JOG bit = 0



MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0000b

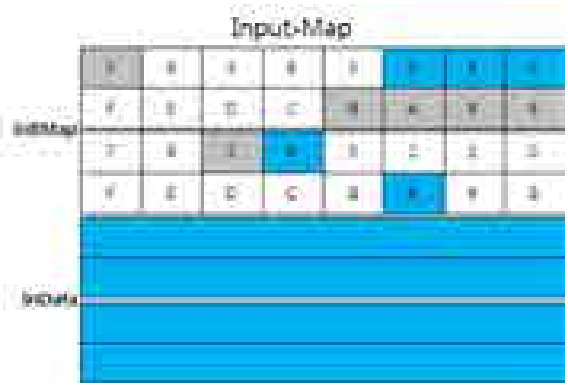
JOG_RESP bit = 1, MOTIONING bit = 1

■ JOG Move – Speed Value Move

조그 운전의 Speed Value Move 는 입력된 값을 실제 속도에 적용하여 조그 운전하는 모션입니다.

✓ **IO-Map 의 명령 및 응답 형태**

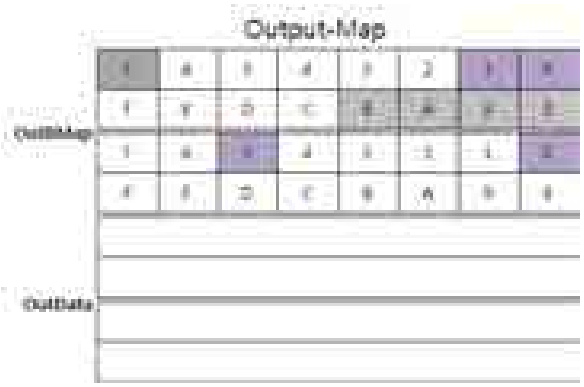
- **-JOG 명령 실행**



MOTION/SETTING bit = 0

CMD_CODE = 0000b , SPD_MODE bit = 1;

+JOG bit = 0, -JOG bit = 1

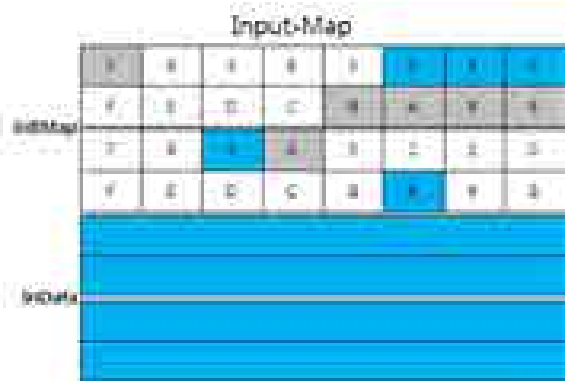


MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0000b

JOG_RESP bit = 1, MOTIONING bit = 1

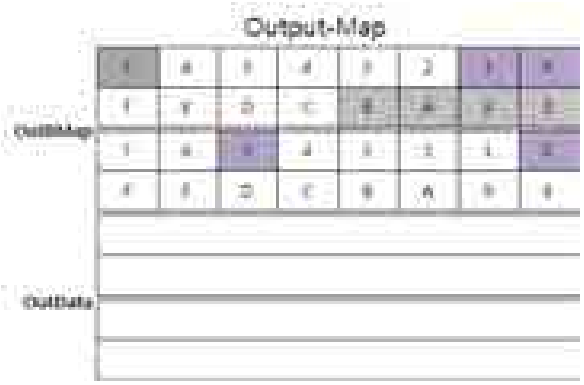
- **+JOG 명령 실행**



MOTION/SETTING bit = 0

CMD_CODE = 0000b , SPD_MODE bit = 1;

+JOG bit = 1, -JOG bit = 0



MOTION/SETTING_RESP bit = 0

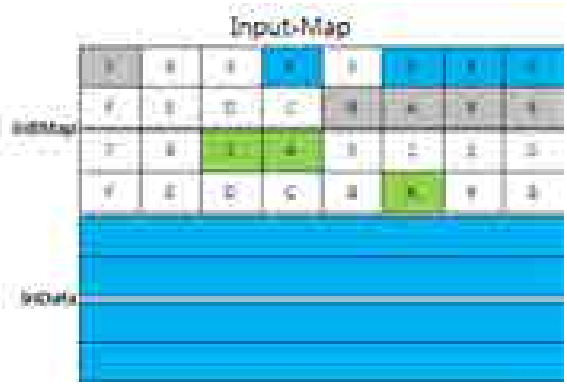
CMD_CODE_RESP = 0000b

JOG_RESP bit = 1, MOTIONING bit = 1

■ JOG Move – Speed Override

조그 운전 중 속도 값을 변경 하는 명령을 Speed Override 명령이라 합니다. 이 명령은 조그 운전 중(+JOG 또는 -JOG 비트가 '1'로 유지된 상태), 변경할 속도값을 입력 후 CMD_START 비트를 '1'로 셋트하여 실행 할 수 있습니다.

✓ **IO-Map 의 명령 및 응답 형태**



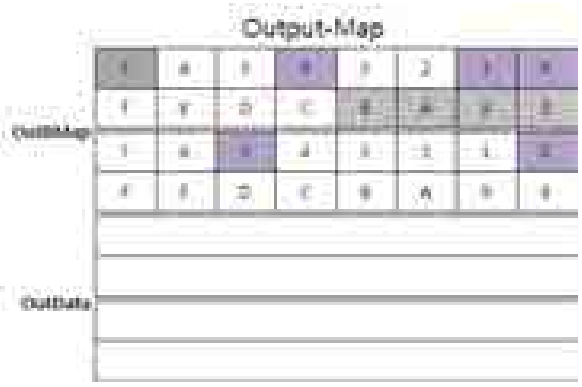
MOTION/SETTING bit = 0

CMD_CODE = 0000b , SPD_MODE bit = 1 or 0;

+JOG bit = 1, -JOG bit = 0

Input-Data = Speed Override Value

CMD_START = 1



MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0000b

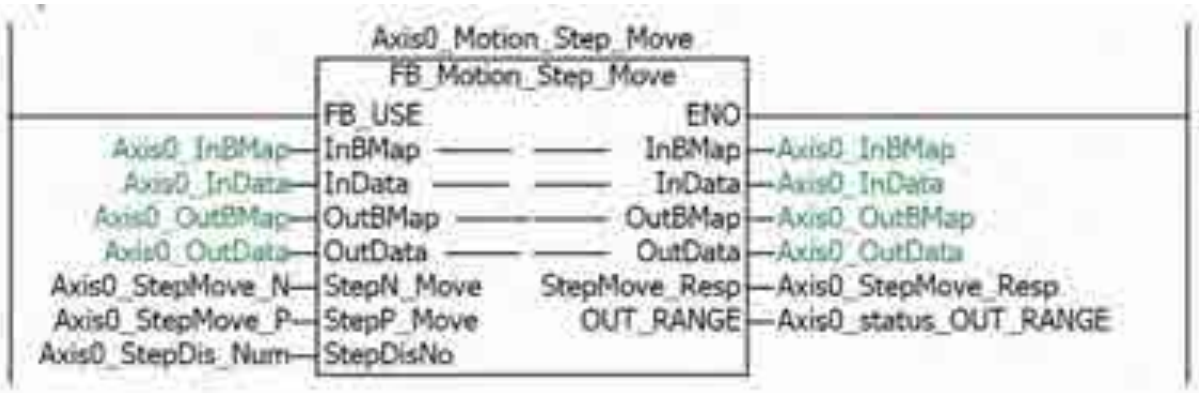
JOG_RESP bit = 1, MOTIONING bit = 1

CMD_START_RESP = 1

6.5.2 스탭 이동

스텝 이동은 모션모드(MOTIONION)상태의 명령 코드(CMD_CODE) '0'에서 동작하며, 스텝이동 거리 번호 0~3 의 값을 지정하여 이동합니다.

Function Block 5. Step 이동



FB_Motion_Step_Move의 사용 변수 목록

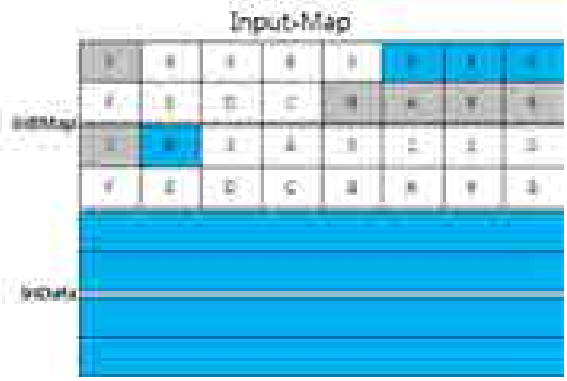
Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DINT	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DINT	No Edge		FALSE	FALSE
StepN_Move	Input	BOOL	No Edge		FALSE	FALSE
StepP_Move	Input	BOOL	No Edge		FALSE	FALSE
StepDisNo	Input	DINT	No Edge		FALSE	FALSE
StepMove_Resp	Output	BOOL	No Edge		FALSE	FALSE
OUT_RANGE	Output	BOOL	No Edge		FALSE	FALSE

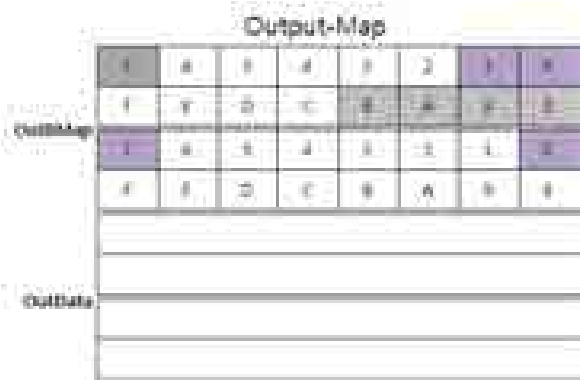
ST 22. Step Move 명령 처리

```
// Step Move Command
IF StepN_Move THEN
    InData:=StepDisNo;
    InBMap:=(InBMap AND DWORD#16#0000F00F) OR DWORD#16#00400000;
ELSIF StepP_Move THEN
    InData:=StepDisNo;
    InBMap:=(InBMap AND DWORD#16#0000F00F) OR DWORD#16#00800000;
ELSE
    InBMap:=InBMap AND DWORD#16#FB3FFFFF;// Step Move bit Clear
END_IF;

//OUT Range Bit
IF (OutBMap AND DWORD#16#00000020)=16#00000020 THEN
    OUT_RANGE := TRUE;
ELSE
    OUT_RANGE := FALSE;
END_IF;

//Step Move Command Response Check
IF (OutBMap AND DWORD#16#00200001) = 16#00800001 THEN
    StepMove_Resp:=TRUE;
ELSE
    StepMove_Resp:=FALSE;
END_IF;
```

✓ **IO-Map 의 명령 및 응답 형태**

- **-SETP Move 명령 실행**





MOTION/SETTING bit = 0

CMD_CODE = 0000b

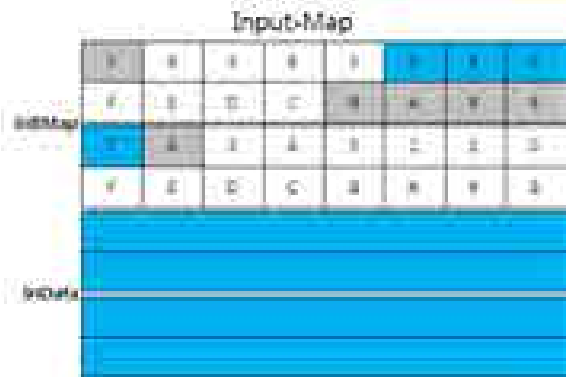
+STEP bit = 0, -STEP bit = 1

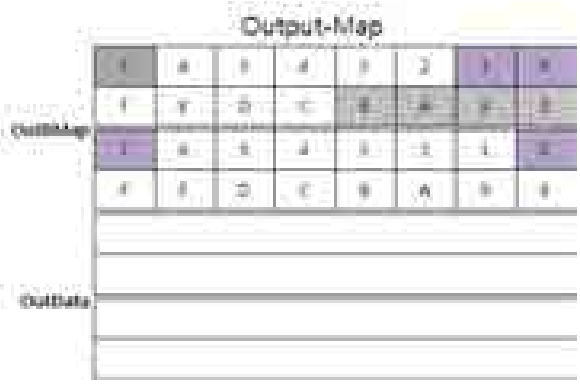
MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0000b

STEP_RESP bit = 1, MOTIONING bit = 1

- **+ SETP Move 명령 실행**





MOTION/SETTING bit = 0

CMD_CODE = 0000b

+STEP bit = 1, -STEP bit = 0

MOTION/SETTING_RESP bit = 0

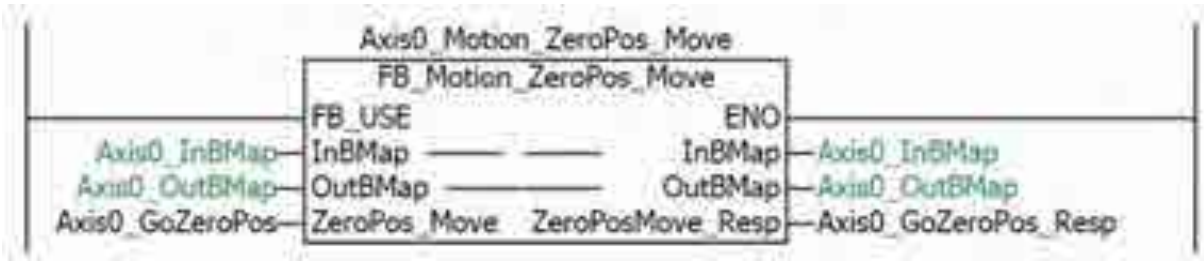
CMD_CODE_RESP = 0000b

STEP_RESP bit = 1, MOTIONING bit = 1

6.5.3 영점 이동

영점 이동은 모션모드(MOTIONION)상태의 명령 코드(CMD_CODE) '0'에서 동작하며, 입력 데이터(D0000)의 값은 무시되며 이동합니다.

Function Block 6. 영점 이동



FB_Motion_ZeroPos_Move의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DINT	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DINT	No Edge		FALSE	FALSE
ZeroPos_Move	Input	BOOL	No Edge		FALSE	FALSE
ZeroPosMove_Resp	Output	BOOL	No Edge		FALSE	FALSE

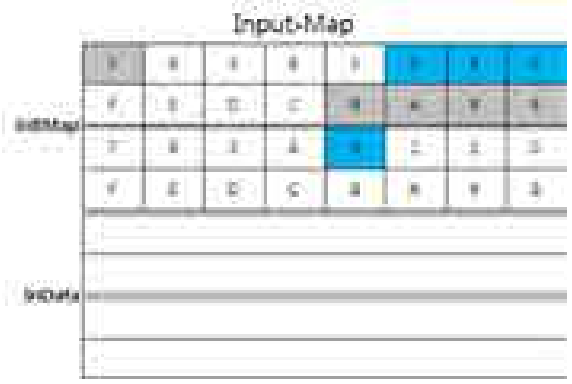
ST 23. Step Move 명령 처리

```
// Step Move Command
IF ZeroPos_Move THEN
    InBMap:=(InBMap AND DWORD#16#0000F0FF) OR DWORD#16#00080000;
ELSE
    InBMap:=InBMap AND DWORD#16#FFF7FFFF;   // zero position Move bit Clear
END_IF;

//Go Zero Position Move Command Response Check
IF (OutBMap AND DWORD#16#00080001) = 16#00080001 THEN
    ZeroPosMove_Resp:=TRUE;
```

```
ELSE
    ZeroPosMove_Resp:=FALSE;
END_IF;
```

✓ **IO-Map 의 명령 및 응답 형태**

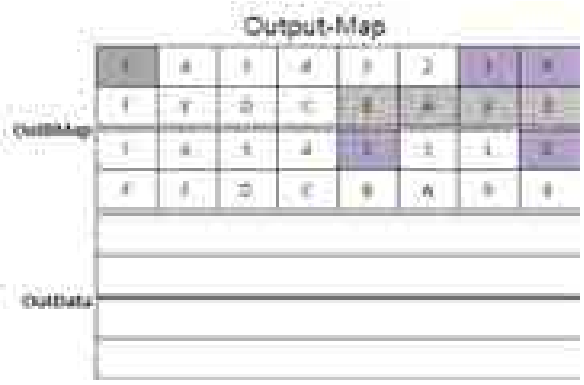
- **영점 이동 명령을 실행**



MOTION/SETTING bit = 0

CMD_CODE = 0000b

GO_ZERO_POS bit = 1

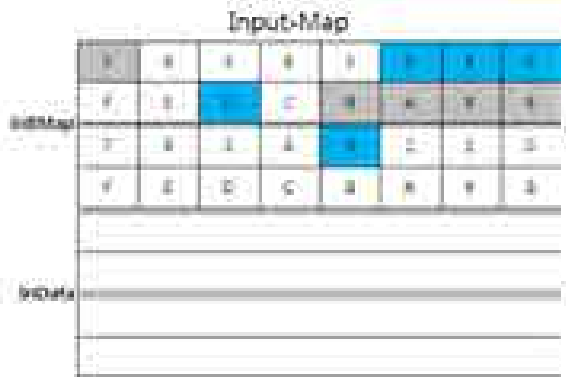


MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0000b

GO_ZERO_POS_RESP bit = 1, MOTIONING bit = 1

- **영점 위치에서 GO_ZERO_POS bit 를 세트 하였을 때**

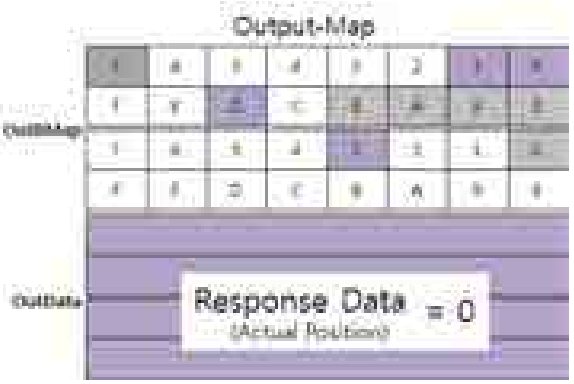


MOTION/SETTING bit = 0

CMD_CODE = 0000b

RESPONSE_TYPE = 0001b

GO_ZERO_POS bit = 0



MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0000b

RESPONSE_TYPE_RESP = 0001b

GO_ZERO_POS_RESP bit = 1,

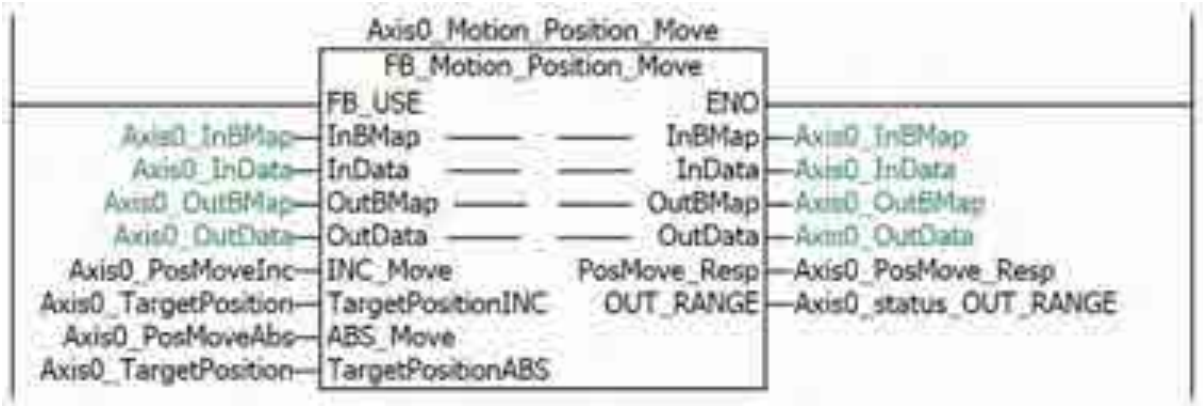
MOTIONING bit = 1

RESPONSE_DATA = Actual Position (0)

6.5.4 위치 이동

상대위치 이동은 모션모드(MOTIONION/SETTING = 0)상태의 명령 코드(CMD_CODE) '1'에서 동작하며, 입력 데이터(D0000)에 위치값으로 상대치 또는 절대치 이동을합니다.

Function Block 7. 기본제어 명령



FB_Motion_Position_Move의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DINT	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DINT	No Edge		FALSE	FALSE
ZeroPos_Move	Input	BOOL	No Edge		FALSE	FALSE
ZeroPosMove_Resp	Output	BOOL	No Edge		FALSE	FALSE
INC_Move	Input	BOOL	No Edge		FALSE	FALSE
TargetPositionINC	Input	DINT	No Edge		FALSE	FALSE
ABS_Move	Input	BOOL	No Edge		FALSE	FALSE
TargetPositionABS	Input	DINT	No Edge		FALSE	FALSE
PosMove_Resp	Input	BOOL	No Edge		FALSE	FALSE
OUT_RANGE	Output	BOOL	No Edge		FALSE	FALSE
cmd_check	Internals	BOOL				

ST 24. Position Move 명령 처리

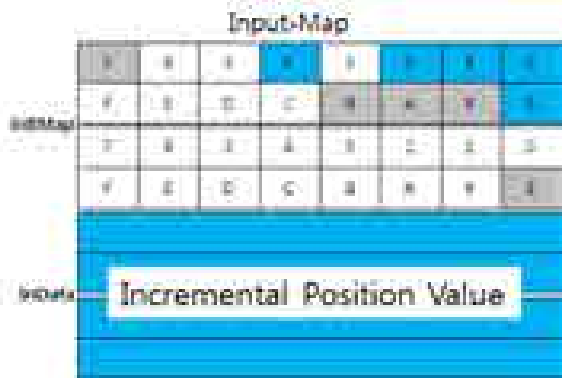
```
// Position Move Command
IF INC_Move THEN
      InData:=TargetPositionINC;
      InBMap:=(InBMap AND DWORD#16#0000F00F) OR DWORD#16#00000110;
      cmd_check:=TRUE;     //cmd_check set

ELSIF ABS_Move THEN
      InData:=TargetPositionABS;
      InBMap:=(InBMap AND DWORD#16#0000F00F) OR DWORD#16#01000110;
      cmd_check:=TRUE;     //cmd_check set

ELSE
      IF cmd_check THEN
            cmd_check := FALSE;
            InBMap:=InBMap AND DWORD#16#FFFFFFEF;   //CMD_START bit Clear
      END_IF;
END_IF;

//OUT Range Bit
IF (OutBMap AND DWORD#16#00000020)=16#00000020 THEN
     OUT_RANGE := TRUE;
ELSE
      OUT_RANGE := FALSE;
END_IF;

//Position Move Command Response Check
IF (OutBMap AND DWORD#16#00000111) = 16#00000111 THEN
      PosMove_Resp:=TRUE;
ELSE
      PosMove_Resp:=FALSE;
END_IF;
```

■ 상대 위치 이동

상대 위치 이동은 입력된 위치 값으로 상대이동을 하는 명령입니다.

✓ **IO-Map 의 명령 및 응답 형태**

- **INC Move 명령 실행**



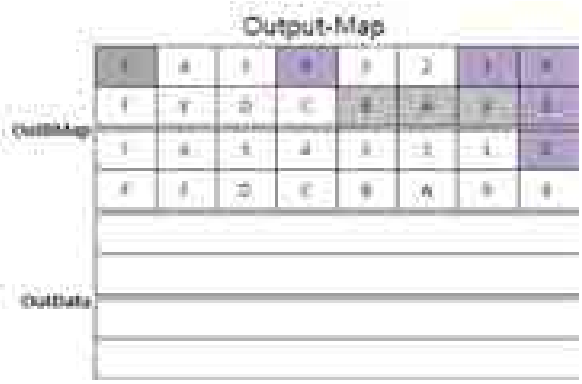
MOTION/SETTING bit = 0

CMD_CODE = 0001b

INC/ABS bit = 0

Command Data = 상대 위치 값

CMD_START bit = 1

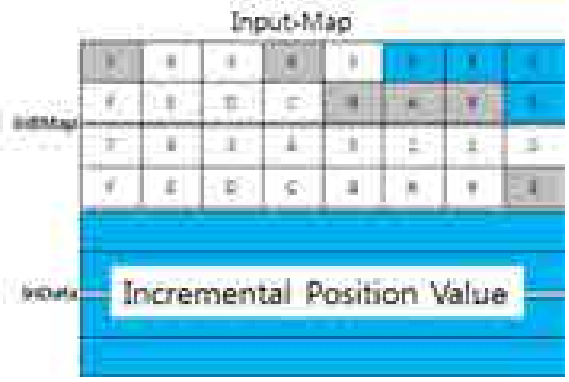


MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0001b

CMD_RESP bit = 1, MOTIONING bit = 1

- **INC Move 명령 후 CMD_START 비트의 해제**



MOTION/SETTING bit = 0

CMD_CODE = 0001b

INC/ABS bit = 0

Command Data = 상대 위치 값

CMD_START bit = 0

MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0001b

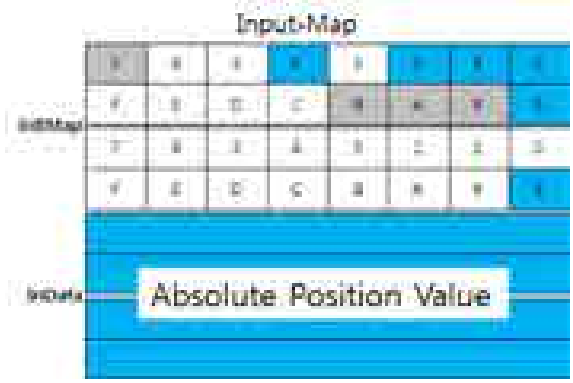
CMD_RESP bit = 0, MOTIONING bit = 1

■ 절대 위치 이동

절대 위치 이동은 입력 데이터(D0000)에 위치값으로 절대치 이동을 합니다.

✓ **IO-Map 의 명령 및 응답 형태**

- **INC Move 명령 실행**



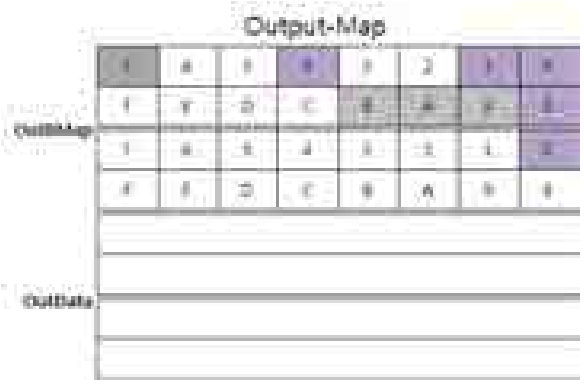
MOTION/SETTING bit = 0

CMD_CODE = 0001b

INC/ABS bit = 1

Command Data = 절대 위치 값

CMD_START bit = 1

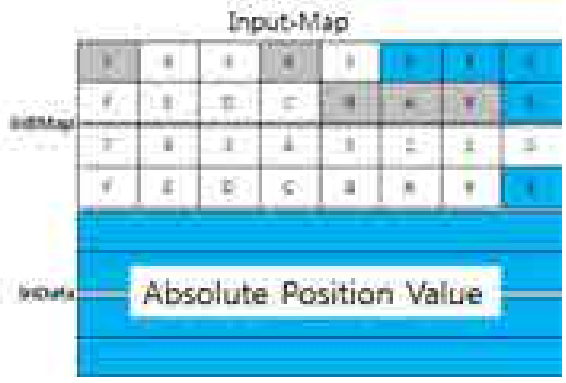


MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0001b

CMD_RESP bit = 1, MOTIONING bit = 1

- **INC Move 명령 후 CMD_START 비트의 해제**



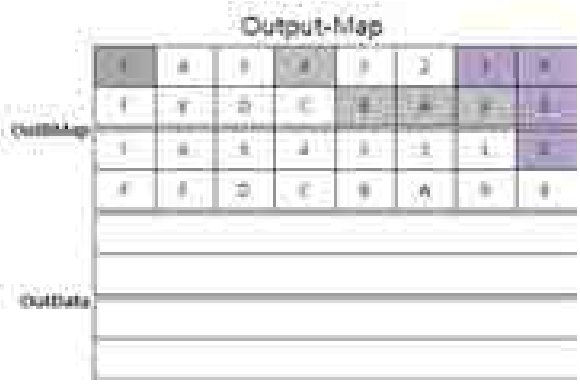
MOTION/SETTING bit = 0

CMD_CODE = 0001b

INC/ABS bit = 1

Command Data = 절대 위치 값

CMD_START bit = 0



MOTION/SETTING_RESP bit = 0

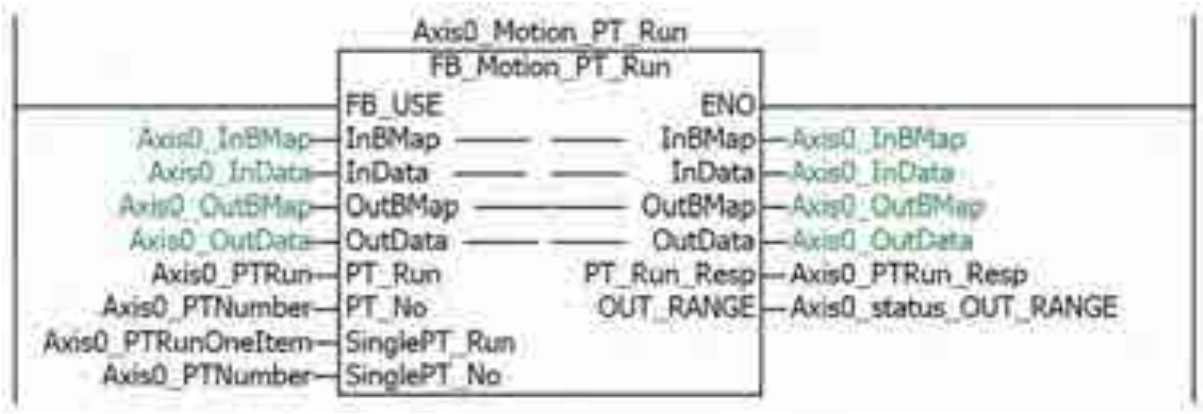
CMD_CODE_RESP = 0001b

CMD_RESP bit = 0, MOTIONING bit = 1

6.5.5 PT 운전

PT 운전은 모션모드(MOTIONION/SETTING = 0)상태의 명령 코드(CMD_CODE) '4'에서 동작하며, 입력 데이터(D0000)에 PT 항목의 번호를 입력하여 운전을 실행 합니다.

Function Block 8. PT 운전



FB_Motion_Motion_PT_Run의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DINT	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DINT	No Edge		FALSE	FALSE
PT_Run	Input	BOOL	No Edge		FALSE	FALSE
PT_No	Input	DINT	No Edge		FALSE	FALSE
SinglePT_Run	Input	BOOL	No Edge		FALSE	FALSE
SinglePT_No	Input	DINT	No Edge		FALSE	FALSE
PT_Run_Resp	Output	BOOL	No Edge		FALSE	FALSE
OUT_RANGE	Output	BOOL	No Edge		FALSE	FALSE
cmd_check	Internals	BOOL				

ST 25. Position Table Run 명령 처리

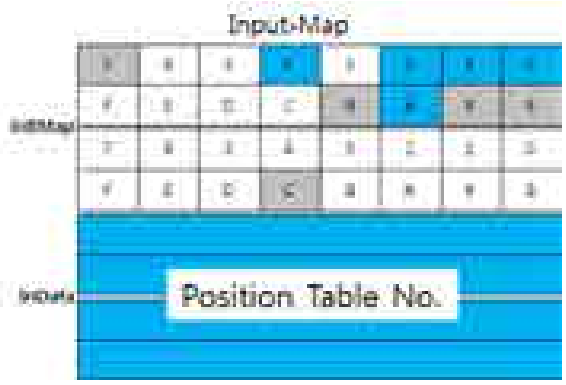
```
// PT Run Command
IF PT_Run THEN
    InData:=PT_No;
    InBMap:=(InBMap AND DWORD#16#0000F00F) OR DWORD#16#00000410;
ELSIF SinglePT_Run THEN
    InData:=SinglePT_No;
    InBMap:=(InBMap AND DWORD#16#0000F00F) OR DWORD#16#10000410;
ELSE
    IF cmd_check THEN
        cmd_check := FALSE;
        InBMap:=InBMap AND DWORD#16#FFFFFFEF;   //CMD_START bit Clear
    END_IF;
END_IF;

//OUT Range Bit
IF (OutBMap AND DWORD#16#00000020)=16#00000020 THEN
    OUT_RANGE := TRUE;
ELSE
    OUT_RANGE := FALSE;
END_IF;

//PT Run Command Response Check
IF (OutBMap AND DWORD#16#00000411) = 16#00000411 THEN
    PT_Run_Resp:=TRUE;
ELSE
    PT_Run_Resp:=FALSE;
END_IF;
```

■ 일반 PT 운전

일반 PT 운전은 입력된 값(D0000)부터 PT 운전을 시작합니다.

✓ **IO-Map 의 명령 및 응답 형태**

- **PT RUN 명령 실행**



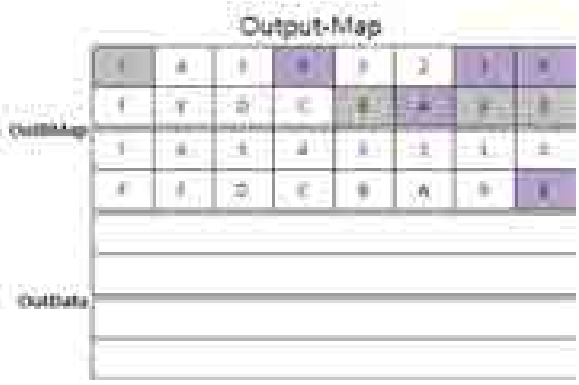
MOTION/SETTING bit = 0

CMD_CODE = 0100b

SINGLE_PT bit = 0

Command Data = PT 번호

CMD_START bit = 1

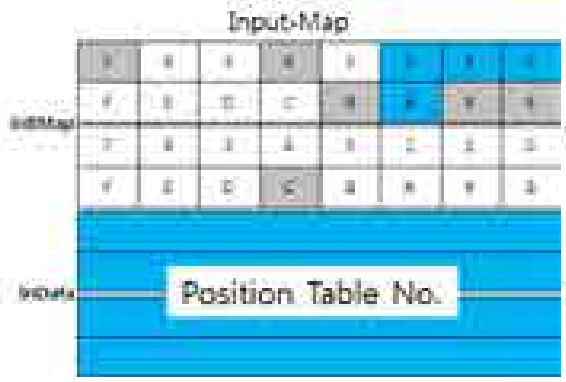


MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0100b

CMD_RESP bit = 1, PT_RUNNING bit = 1

- **PT RUN 명령 후 CMD_START 비트의 해제**



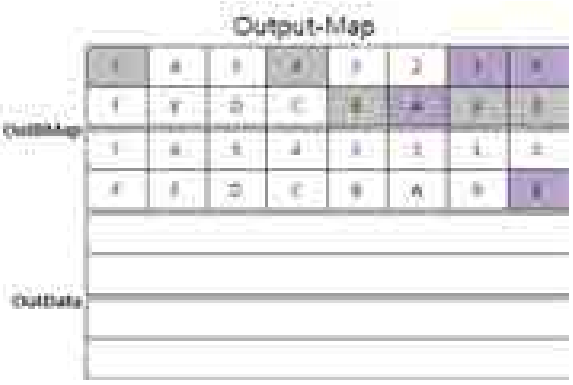
MOTION/SETTING bit = 0

CMD_CODE = 0100b

SINGLE_PT bit = 0

Command Data = PT 번호

CMD_START bit = 0



MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0100b

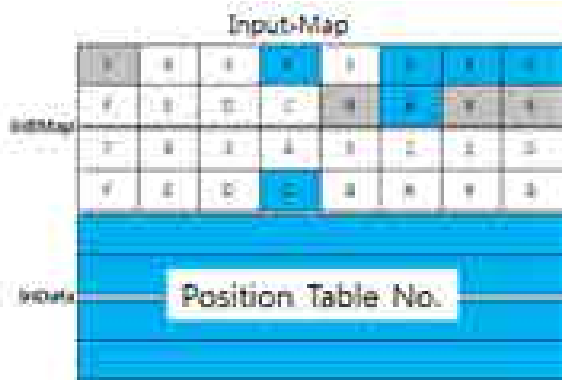
CMD_RESP bit = 0, PT_RUNNING bit = 1

■ 싱글 PT 운전

싱글 PT 운전은 입력된 값(D0000)부터 하나의 PT 항목에 대하여 운전합니다.

✓ **IO-Map 의 명령 및 응답 형태**

- **싱글 PT RUN 명령 실행**



MOTION/SETTING bit = 0

CMD_CODE = 0100b

SINGLE_PT bit = 1

Command Data = PT 번호

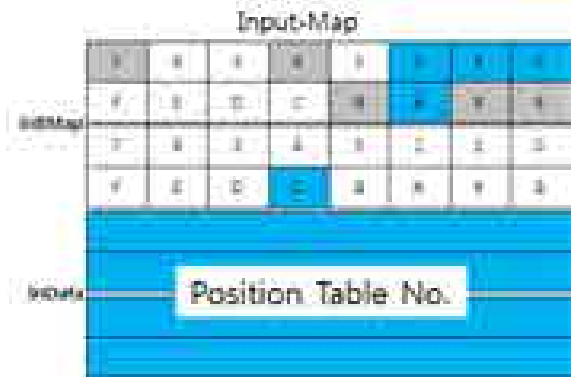
CMD_START bit = 1

MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0100b

CMD_RESP bit = 1, PT_RUNNING bit = 1

- **싱글 PT RUN 명령 후 CMD_START 비트의 해제**



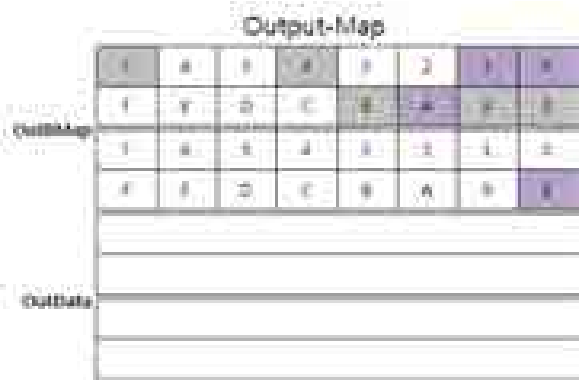
MOTION/SETTING bit = 0

CMD_CODE = 0100b

SINGLE_PT bit = 1

Command Data = PT 번호

CMD_START bit = 0



MOTION/SETTING_RESP bit = 0

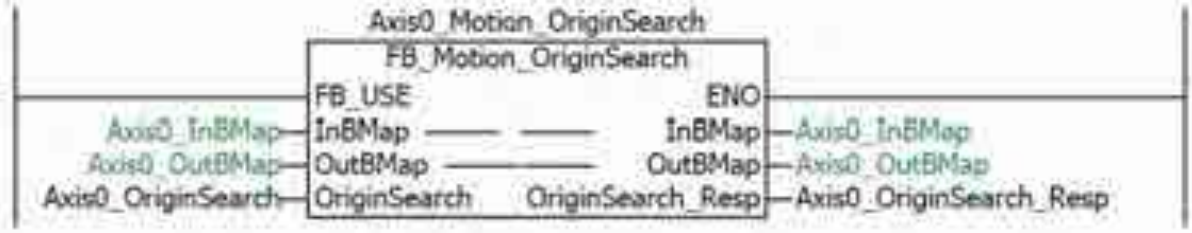
CMD_CODE_RESP = 0100b

CMD_RESP bit = 0, PT_RUNNING bit = 1

6.5.6 원점 이동

원점 이동 명령은 모션모드(MOTIONION/SETTING = 0)상태의 명령 코드(CMD_CODE) '7'에서 동작하며, 입력 데이터(D0000)에 입력된 값과 무관하게 동작합니다.

Function Block 9. 원점 이동



FB_Motion_OriginSearch의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OriginSearch	Input	BOOL	No Edge		FALSE	FALSE
OriginSearch_Resp	Output	BOOL	No Edge		FALSE	FALSE
cmd_check	Internals	BOOL				

ST 26. Origin Search 명령 처리

```
// Origin Search Command
IF OriginSearch THEN
    InBMap:=(InBMap AND DWORD#16#0000F00F) OR DWORD#16#00000710;
    cmd_check := TRUE;
ELSE
    IF cmd_check THEN
        cmd_check := FALSE;
        InBMap:=InBMap AND DWORD#16#FFFFFFEF;   //CMD_START bit Clear
    END_IF;
END_IF;
```

```
// Origin Search Command Response Check
IF (OutBMap AND DWORD#16#00000711) = 16#00000711 THEN
      OriginSearch_Resp:=TRUE;
ELSE
      OriginSearch_Resp:=FALSE;
END_IF;
```

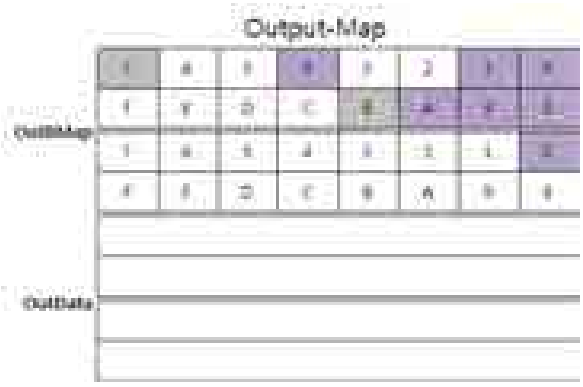
✓ **IO-Map 의 명령 및 응답 형태**

- **원점 이동 실행**

MOTION/SETTING bit = 0

CMD_CODE = 0111b

CMD_START bit = 1



MOTION/SETTING_RESP bit = 0

CMD_CODE_RESP = 0111b

CMD_RESP bit = 1, PT_RUNNING bit = 1

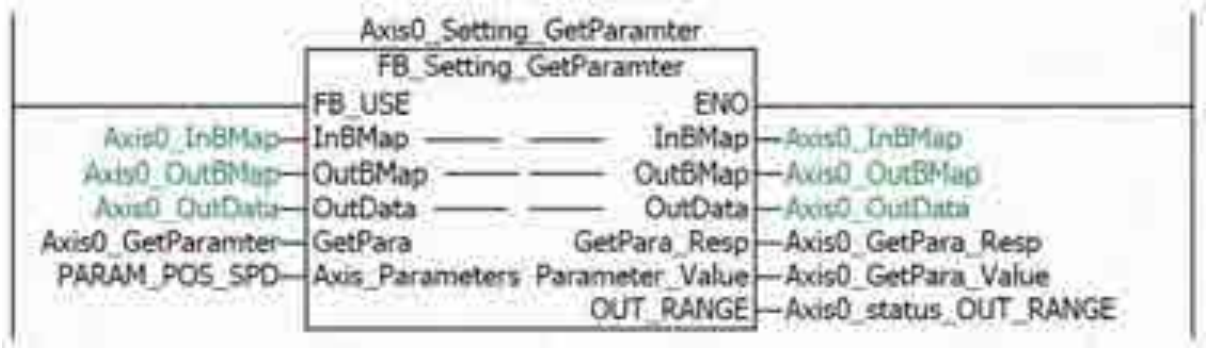
6.6 파라미터 설정

파라미터 설정은 해당 축의 모터 드라이브에 대한 파라미터와 PT 정보의 확인 및 설정, 저장을 할 수 있습니다.

6.6.1 파라미터 확인

파라미터 확인은 설정모드(MOTIONION/SETTING=1)상태의 명령 코드(CMD_CODE) '8' 에서 동작하며, 인덱스 영역(I/O Bit Map 의 상위 WORD 영역)에 입력된 파라미터 번호에 대한 값을 요청하는 명령 입니다. 이때 입력 데이터(Input Data)에 입력된 값은 무관합니다.

Function Block 10. 파라미터 설정



FB_Setting_GetParameter의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DWORD	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DWORD	No Edge		FALSE	FALSE
GetPara	Input	BOOL	No Edge		FALSE	FALSE
Axis_Parameters	Input	PARAMETERS	No Edge		FALSE	FALSE
GetPara_Resp	Output	BOOL	No Edge		FALSE	FALSE

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
Parameter_Value	Output	DINT	No Edge		FALSE	FALSE
OUT_RANGE	Output	BOOL	No Edge		FALSE	FALSE
cmd_check	Internals	BOOL				

본 매뉴얼의 펑션블럭 FB_Setting_GetParameter 의 Axis_parameters 에 입력되는 값은 ENUM 으로 선언 하여 사용합니다. (OMRON SYSMAC STUDIO 에서는 ENUM 선언 시 10 진수 입력만 가능합니다.)

데이터 타입 - Enumerated

Type Name	Member Name	Enum Value	Comment
PARAMETERS			열거 유형
	PARAM_JOGSTEP0	256	Pn#0100 PARAM_JOGSTEP0
	PARAM_JOGSTEP1	257	Pn#0101 PARAM_JOGSTEP1
	PARAM_JOGSTEP2	258	Pn#0102 PARAM_JOGSTEP2
	PARAM_JOGSTEP3	259	Pn#0103 PARAM_JOGSTEP3
	PARAM_JOG_RATIO	260	Pn#0104 PARAM_JOG_RATIO
	PARAM_JOG_RATIO_SPD	261	Pn#0105 PARAM_JOG_RATIO_SPD
	PARAM_STEP_DIST0	512	Pn#0200 PARAM_STEP_DIST0
	PARAM_STEP_DIST1	513	Pn#0201 PARAM_STEP_DIST1
	PARAM_STEP_DIST2	514	Pn#0202 PARAM_STEP_DIST2
	PARAM_STEP_DIST3	515	Pn#0203 PARAM_STEP_DIST3
	PARAM_STEP_SPD	516	Pn#0204 PARAM_STEP_SPD
	PARAM_POS_SPD	1024	Pn#0400 PARAM_POS_SPD
	PARAM_MOTORLEAD	2034	Pn#0900 PARAM_MOTORLEAD
	PARAM_GEARRATIO	2035	Pn#0901 PARAM_GEARRATIO
	PARAM_DECIMAL	2036	Pn#0902 PARAM_DECIMAL
	PARAM_PULSE_PER_REVOL	2560	Pn#0A00 Pulse Per Revolution

Type Name	Member Name	Enum Value	Comment
	PARAM_MAXSPEED	2561	Pn#0A01 Axis Max Speed
	PARAM_STARTSPEED	2562	Pn#0A02 Axis Start Speed
	PARAM_ACCTIME	2563	Pn#0A03 Axis Acc Time
	PARAM_DECTIME	2564	Pn#0A04 Axis Dec Time
	PARAM_SPD_OVERRIDE	2565	Pn#0A05 Speed Override
	PARAM_JOG_SPD	2566	Pn#0A06 Jog Speed
	PARAM_JOGSTART_SPD	2567	Pn#0A07 Jog Start Speed
	PARAM_JOG_ACCDEC_TIME	2568	Pn#0A08 Jog Acc Dec Time
	PARAM_ALARMLOGIC	2569	Pn#0A09 Servo Alarm Logic
	PARAM_SVN_LOGIC	2570	Pn#0A0A Servo On Logic
	PARAM_SERVO_ALMRST_LOGIC	2571	Pn#0A0B Servo Alarm Reset Logic
	PARAM_STEPRUNLOGIC	2572	Pn#0A0C Step Run/Stop Logic
	PARAM_SETPARMRSTLOGIC	2573	Pn#0A0D Step Alarm Reset Logic
	PARAM_SWLIMIT_PLUS	2574	Pn#0A0E S/W Limit Plus Value
	PARAM_SWLIMIT_MINUS	2575	Pn#0A0F S/W Limit Minus Value
	PARAM_SWLIMIT_STOP_METHOD	2576	Pn#0A10 S/W Limit Stop Method
	PARAM_HWLIMIT_STOP_METHOD	2577	Pn#0A11 H/W Limit Stop Method
	PARAM_LIMIT_LOGIC	2578	Pn#0A12 Limit Sensor Logic
	PARAM_ORG_SPEED	2579	Pn#0A13 Org Speed
	PARAM_ORGSEARCH_SPD	2580	Pn#0A14 Org Search Speed
	PARAM_ORG_ACCDEC	2581	Pn#0A15 Org Acc Dec Time
	PARAM_ORG_METHHOD	2582	Pn#0A16 Org Method
	PARAM_ORG_DIR	2583	Pn#0A17 Org Dir
	PARAM_ORG_OFFSET	2584	Pn#0A18 Org OffSet
	PARAM_ORG_POSITION_SET	2585	Pn#0A19 Org Position Set
	PARAM_ORG_SENSOR_LOGIC	2586	Pn#0A1A Org Sensor Logic
	PARAM_POS_LOOP_GAIN	2587	Pn#0A1B Position Loop Gain

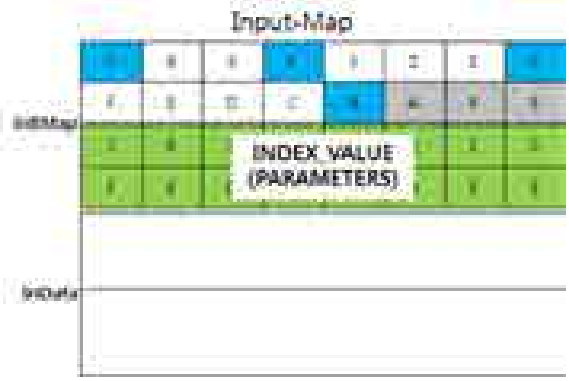
Type Name	Member Name	Enum Value	Comment
	PARAM_STOP_CURRENT	2588	Pn#0A1C Stop Current
	PARAM_INP_VALUE	2589	Pn#0A1D Inpos Value
	PARAM_POSTRACKING	2590	Pn#0A1E Pos Tracking Limit
	PARAM_MOTION_DIR	2591	Pn#0A1F Motion Dir
	PARAM_LIMIT_SENSOR_DIR	2592	Pn#0A20 Limit Sensor Dir
	PARAM_ORG_TORQUE_RATIO	2593	Pn#0A21 Org Torque Ratio
	PARAM_ENCODER_MULTIPLY	2594	Pn#0A22 Encoder Multiply Value
	PARAM_POS_OVERFLOW	2595	Pn#0A23 Pos. Error Overflow Limit
PTITEM			열거 유형
	PT_COMMAND	4906	Pn#1000 + #n PT Command
	PT_POSITION	4608	Pn#1200 + #n PT Position
	PT_START_SPEED	5120	Pn#1400 + #n PT Start Speed
	PT_MOVE_SPEED	5632	Pn#1600 + #n PT Move Speed
	PT_ACC_TIME	6144	Pn#1800 + #n PT Accel Time
	PT_DEC_TIME	6656	Pn#1A00 + #n PT Decel Time
	PT_JUMP_TABLE_NUM	7168	Pn#1C00 + #n PT Jump Table No.
	PT_JUMP_PT0	7680	Pn#1E00 + #n PT Jump PT0
	PT_JUMP_PT1	8192	Pn#2000 + #n PT Jump PT1
	PT_JUMP_PT2	8704	Pn#2200 + #n PT Jump PT2
	PT_LOOP_COUNT	9216	Pn#2400 + #n PT Loop Count
	PT_LOOP_JUMP_TABLE_NUM	9728	Pn#2600 + #n PT Loop Jump Table No.
	PT_PT_SET	10240	Pn#2800 + #n PT PT Set
	PT_LOOP_COUNT_CLEAR	10752	Pn#2A00 + #n PT Loop count Clear
	PT_WAIT_TIME	11264	Pn#2C00 + #n PT Wait Time
	PT_CHECK_INP	11776	Pn#2E00 + #n PT Check Inposition
	PT_CONTIUOUS_ACTION	12288	Pn#3000 + #n PT Contiuous Action

ST 27. Get Parameter 명령 처리

```
// Get Parameter Command
IF GetPara THEN
    //Input INEX Value
    Axis_ParameterNo := DINT_TO_DWORD(Axis_Parameters);
    InBMap := InBMap AND DWORD#16#0000FFFF;
    InBMap := InBMap OR SHL(Axis_ParameterNo,16);// Index Value to Input Map
    // Get Parameter bit Set
    InBMap:=(InBMap AND DWORD#16#FFFF000F) OR DWORD#16#00000890;
    cmd_check := TRUE;
ELSE
    IF cmd_check THEN
        cmd_check := FALSE;
        InBMap:=InBMap AND DWORD#16#FFFFFFEF;   //CMD_START bit Clear
    END_IF;
END_IF;

//OUT Range Bit
IF (OutBMap AND DWORD#16#00000020)=16#00000020 THEN
    OUT_RANGE := TRUE;
ELSE
    OUT_RANGE := FALSE;
END_IF;

// Get Parameter Command Response Check
// CONNECT bit & CMD_Start & Setting & CMD_Code
IF (OutBMap AND DWORD#16#00000891)=16#00000891 THEN
    IF (OutBMap AND SHL(Axis_ParameterNo,16))=SHL(Axis_ParameterNo,16) THEN
        Parameter_Value := OutData;
        GetPara_Resp := TRUE;
    ELSE
        GetPara_Resp := FALSE;
    END_IF;
ELSE
    GetPara_Resp := FALSE;
END_IF;
```

✓ **IO-Map 의 명령 및 응답 형태**

- **원점 이동 실행**

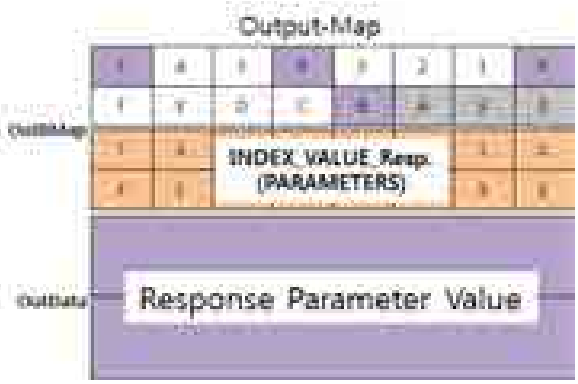


MOTION/SETTING bit = 1

CMD_CODE = 1000b

INDEX_VALUE = Index Value (Parameter No)

CMD_START bit = 1



MOTION/SETTING_RESP bit = 1

CMD_CODE_RESP = 1000b

INDEX_VALUE_RESP = Index Value (Parameter No)

CMD_RESP bit = 1

6.6.2 파라미터 변경

파라미터 변경은 설정모드(MOTIONION/SETTING=1)상태의 명령 코드(CMD_CODE) '9' 에서 동작하며, 인덱스 영역(I/O Bit Map 의 상위 WORD 영역)에 작성된 파라미터 번호에 입력 데이터영역(D0000)에 지정된 값으로 변경하는 명령입니다.

Function Block 11. 파라미터 변경

FB_Setting_SetParameter의 사용 변수 목록

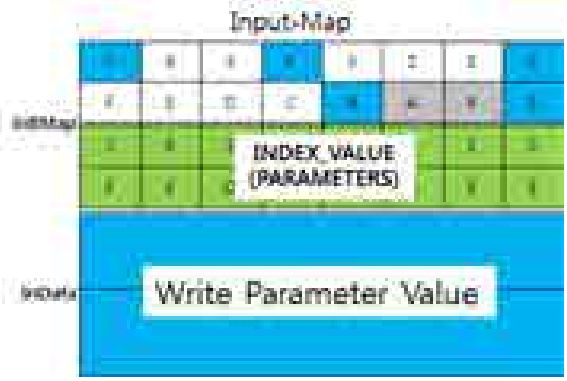
Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DWORD	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DWORD	No Edge		FALSE	FALSE
SetPara	Input	BOOL	No Edge		FALSE	FALSE
Axis_Parameters	Input	PARAMETERS	No Edge		FALSE	FALSE
Parameter_Value	Input	DINT	No Edge		FALSE	FALSE
SetPara_Resp	Output	BOOL	No Edge		FALSE	FALSE
Parameter_Resp_Value	Output	DINT	No Edge		FALSE	FALSE
OUT_RANGE	Output	BOOL	No Edge		FALSE	FALSE
cmd_check	Internals	BOOL				

ST 28. Set Parameter 명령 처리

```
// Set Parameter Command
IF SetPara THEN
    //Input INEX Value
    Axis_ParameterNo := DINT_TO_DWORD(Axis_Parameters);
    InBMap := InBMap AND DWORD#16#0000FFFF;
    InBMap := InBMap OR SHL(Axis_ParameterNo,16);// Index Value to Input Map
    // Get Parameter
    InData := Parameter_Value;
    InBMap:=(InBMap AND DWORD#16#FFFF000F) OR DWORD#16#00000990;
    cmd_check := TRUE;
ELSE
    IF cmd_check THEN
        cmd_check := FALSE;
        InBMap:=InBMap AND DWORD#16#FFFFFFEF; //CMD_START bit Clear
    END_IF;
END_IF;

//OUT Range Bit
IF (OutBMap AND DWORD#16#00000020)=16#00000020 THEN
    OUT_RANGE := TRUE;
ELSE
    OUT_RANGE := FALSE;
END_IF;

// Set Parameter Command Response Check
// CONNECT bit & CMD_Start & Setting & CMD_Code
IF (OutBMap AND DWORD#16#00000991)=16#00000991 THEN
    IF (OutBMap AND SHL(Axis_ParameterNo,16))=SHL(Axis_ParameterNo,16) THEN
        Parameter_Resp_Value := OutData;
        SetPara_Resp := TRUE;
    ELSE
        SetPara_Resp := FALSE;
    END_IF;
ELSE
    SetPara_Resp := FALSE;
END_IF;
```

✓ **IO-Map 의 명령 및 응답 형태**

- **파라미터 값 변경 실행**



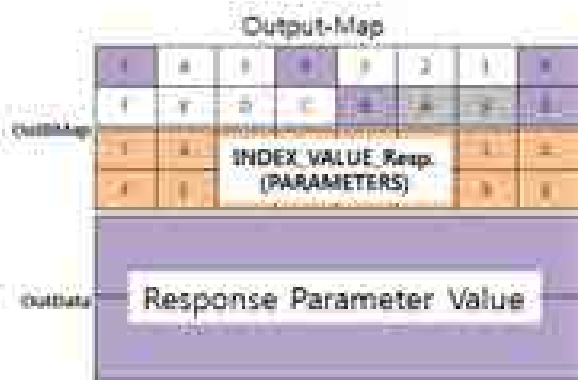
MOTION/SETTING bit = 1

CMD_CODE = 1001b

INDEX_VALUE = Index Value (Parameter No)

COMMAND_WORD_DATA = Write Parameter Value

CMD_START bit = 1



MOTION/SETTING_RESP bit = 1

CMD_CODE_RESP = 1001b

INDEX_VALUE_RESP = Index Value (Parameter No)

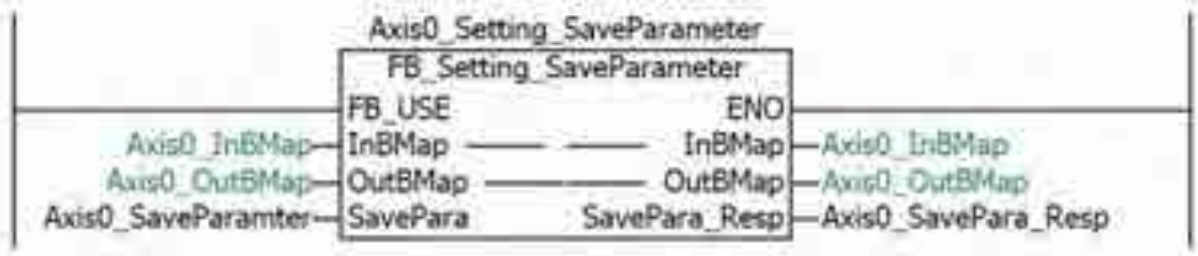
RESPONSE_DATA = Response Parameter Value

CMD_RESP bit = 1

6.6.3 파라미터 저장

파라미터 저장은 설정모드(MOTIONION/SETTING=1)상태의 명령 코드(CMD_CODE) '14' 에서 동작하여 ROM 영역에 파라미터 값을 저장 합니다.

Function Block 12. 파라미터 저장



FB_Setting_SaveParameter의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
SavePara	Input	BOOL	No Edge		FALSE	FALSE
SavePara_Resp	Output	BOOL	No Edge		FALSE	FALSE
cmd_check	Internals	BOOL				

ST 29. Save Parameter 명령 처리

```
// Get Parameter Command
IF SavePara THEN
    //Input INEX Value
    InBMap := InBMap AND DWORD#16#0000FFFF;
    InBMap:=(InBMap AND DWORD#16#FFFF000F) OR DWORD#16#00000E90;
    cmd_check := TRUE;
```

```
ELSE
      IF cmd_check THEN
            cmd_check := FALSE;
            InBMap:=InBMap AND DWORD#16#FFFFFFEF;   //CMD_START bit Clear
      END_IF;
END_IF;

// Save Parameter Command Response Check
// CONNECT bit & CMD_Start & Setting & CMD_Code
IF (OutBMap AND DWORD#16#00000E91)=16#00000E91 THEN
      SavePara_Resp := TRUE;
ELSE
      SavePara_Resp := FALSE;
END_IF;
```

✓ **IO-Map 의 명령 및 응답 형태**

- **파라미터 저장 명령**

MOTION/SETTING bit = 1

CMD_CODE = 1110b

CMD_START bit = 1

MOTION/SETTING_RESP bit = 1

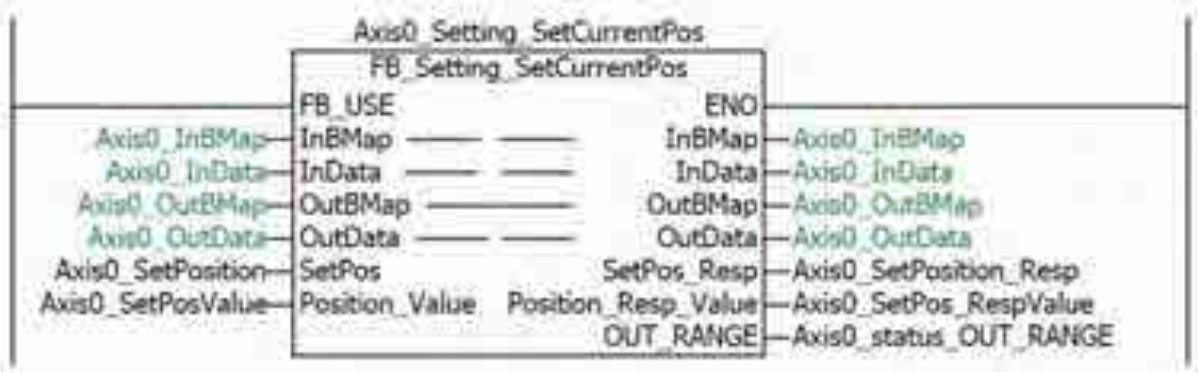
CMD_CODE_RESP = 1110b

CMD_RESP bit = 1

6.7 위치 지정

위치 지정 명령은 설정모드(MOTIONION/SETTING=1)상태의 명령 코드(CMD_CODE) '10' 에서 동작하여 추종중인 위치 값(Command Position)을 변경합니다.

Function Block 13. 위치 지정



FB_Setting_SetCurrentPos의 사용 변수 목록

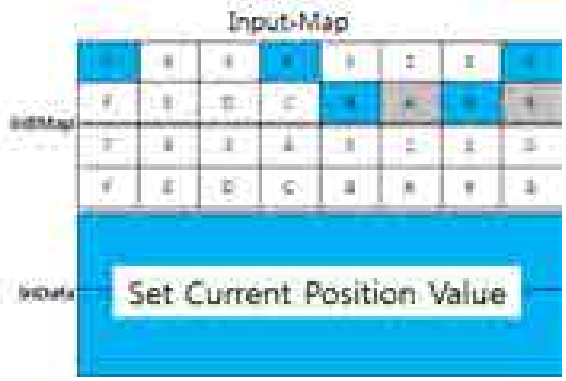
Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
InData	In/Out	DWORD	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DWORD	No Edge		FALSE	FALSE
SetPos	Input	BOOL	No Edge		FALSE	FALSE
Position_Value	Input	DINT	No Edge		FALSE	FALSE
SetPos_Resp	Output	BOOL	No Edge		FALSE	FALSE
Position_Resp_Value	Output	DINT	No Edge		FALSE	FALSE
cmd_check	Internals	BOOL				

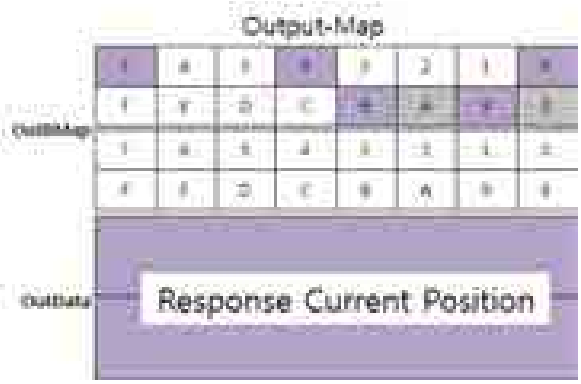
ST 30. Set Current Position 명령 처리

```
// Set Current Position Command
IF SetPos THEN
    // Set Position
    InData := Position_Value;
    InBMap:=(InBMap AND DWORD#16#0000000F) OR DWORD#16#00000A90;
    cmd_check := TRUE;
ELSE
    IF cmd_check THEN
        cmd_check := FALSE;
        InBMap:=InBMap AND DWORD#16#FFFFFFEF;   //CMD_START bit Clear
    END_IF;
END_IF;

//OUT Range Bit
IF (OutBMap AND DWORD#16#00000020)=16#00000020 THEN
    OUT_RANGE := TRUE;
ELSE
    OUT_RANGE := FALSE;
END_IF;

// Set Current Position Command Response Check
// CONNECT bit & CMD_Start & Setting & CMD_Code
IF (OutBMap AND DWORD#16#00000A91)=16#00000A91 THEN
    IF InData=OutData THEN
        Position_Resp_Value := OutData;
        SetPos_Resp := TRUE;
    ELSE
        SetPos_Resp := FALSE;
    END_IF;
ELSE
    SetPos_Resp := FALSE;
END_IF;
```

✓ **IO-Map 의 명령 및 응답 형태**

- **현재 위치 지정 명령**





MOTION/SETTING bit = 1

CMD_CODE = 1010b

COMMAND_WORD_DATA = Set Current Position

CMD_START bit = 1

MOTION/SETTING_RESP bit = 1

CMD_CODE_RESP = 1010b

RESPONSE_DATA = Response Current Position

CMD_RESP bit = 1

NOTE 1: 위치 지정 명령을 실행하면, Output Data 에는 현재 위치 정보가 표시 됩니다.

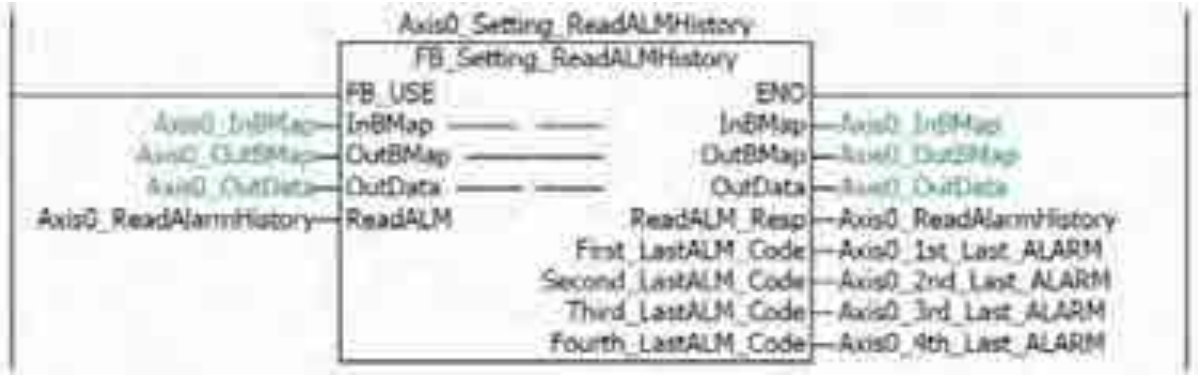
6.8 알람 내역

모션게이트는 최근에 발생한 알람부터 4개에 대한 알람내역을 관리 할 수 있습니다.

6.8.1 알람 내역 확인

알람 내역 요청은 설정모드(MOTIONION/SETTING=1)상태의 명령 코드(CMD_CODE) '12' 에서 동작하여 해당 축에 발생한 알람 내역(Alarm History)를 확인 할 수 있습니다.

Function Block 14. 알람 내역 요청



FB_Setting_ReadALMHistory의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DWORD	No Edge		FALSE	FALSE
ReadALM	Input	BOOL	No Edge		FALSE	FALSE
ReadALM_Resp	Output	BOOL	No Edge		FALSE	FALSE
First_LastALM_Code	Output	BYTE	No Edge		FALSE	FALSE
Second_LastALM_Code	Output	BYTE	No Edge		FALSE	FALSE
Third_LastALM_Code	Output	BYTE	No Edge		FALSE	FALSE
Fourth_LastALM_Code	Output	BYTE	No Edge		FALSE	FALSE

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
cmd_check	Internals	BOOL				
AlarmHistory	Internals	ALARM_HISTORY	FALSE		FALSE	

Output Data 로 수신되는 데이터는 32 비트 정수로 수신 됩니다. 수신된 데이터는 8 비트 BYTE 형식으로 4 개로 분할됩니다. 이는 열거 형(UNION) 데이터 구조로 나누어 사용할 수 있습니다. 본 매뉴얼에서 사용된 데이터 유형은 다음과 같습니다.

데이터 타입 - Union

Type Name	Member Name	Base Type	Comment
ALARM_HISTORY		UNION	열거 유형
	Alarm_Buffer	DWORD	위치추종 데이터
	AlarmCodes	ARRAY[0..3] OF BYTE	현재 위치 데이터

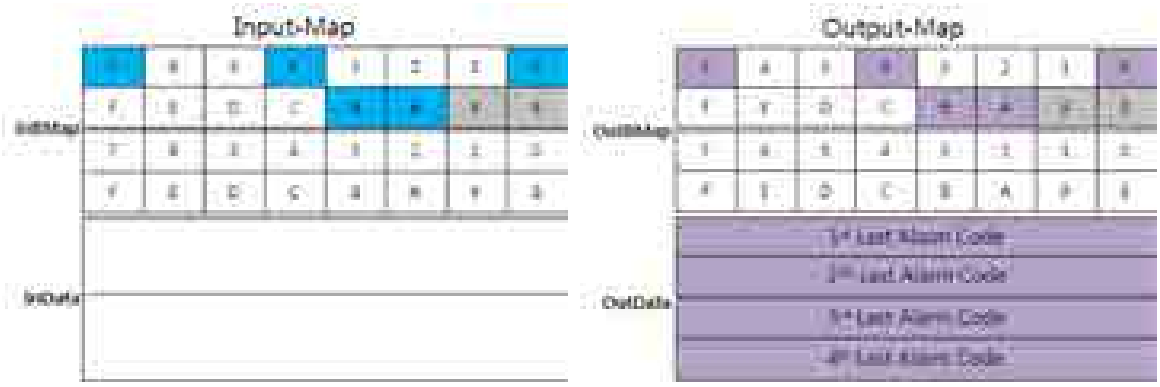
ST 31. Read Alarm History 명령 처리

```
// Read Alarm History Command
IF ReadALM THEN
    // Get Alarm History
    InBMap:=(InBMap AND DWORD#16#0000000F) OR DWORD#16#00000A90;
    cmd_check := TRUE;
ELSE
    IF cmd_check THEN
        cmd_check := FALSE;
        InBMap:=InBMap AND DWORD#16#FFFFFFEF;   //CMD_START bit Clear
    END_IF;
END_IF;

// Read Alarm History Command Response Check
// CONNECT bit & CMD_Start & Setting & CMD_Code
IF (OutBMap AND DWORD#16#00000A91)=16#00000A91 THEN
    AlarmHistory.Alarm_Buffer:= DINT_TO_DWORD(OutData);
    First_LastALM_Code := AlarmHistory.AlarmCodes[3];
    Second_LastALM_Code:= AlarmHistory.AlarmCodes[2];
    Third_LastALM_Code:= AlarmHistory.AlarmCodes[1];
```

```
    Fourth_LastALM_Code:= AlarmHistory.AlarmCodes[0];
    ReadALM_Resp := TRUE;
ELSE
    ReadALM_Resp := FALSE;
END_IF;
```

✓ **IO-Map 의 명령 및 응답 형태**

- **알람 내역 확인 명령 실행**



MOTION/SETTING bit = 1

CMD_CODE = 1100b

CMD_START bit = 1

MOTION/SETTING_RESP bit = 1

CMD_CODE_RESP = 1100b

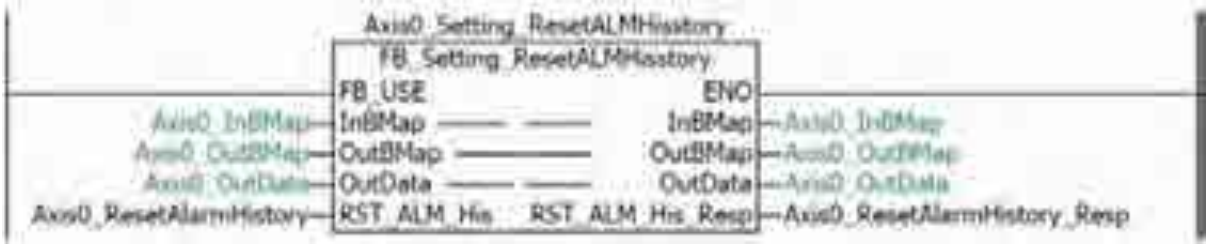
CMD_RESP bit = 1

RESPONSE_DATA = Alarm History

6.8.2 알람 내역 초기화

알람 내역 초기화는 설정모드(MOTIONION/SETTING=1)상태의 명령 코드(CMD_CODE) '13' 에서 동작하여 해당 축에 발생한 알람 내역(Alarm History)를 확인 할 수 있습니다.

Function Block 15. 알람 내역 초기화



FB_Setting_ResetALMHistory의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DWORD	No Edge		FALSE	FALSE
RST_ALM_His	Input	BOOL	No Edge		FALSE	FALSE
RST_ALM_His_Resp	Output	BOOL	No Edge		FALSE	FALSE
cmd_check	Internals	BOOL				

ST 32. Reset Alarm History 명령 처리

```
// Reset Alarm History Command
IF RST_ALM_His THEN
    //Input INEX Value
    InBMap := InBMap AND DWORD#16#0000FFFF;
    InBMap:=(InBMap AND DWORD#16#FFFF000F) OR DWORD#16#00000D90;
    cmd_check := TRUE;
```

```
ELSE
      IF cmd_check THEN
            cmd_check := FALSE;
            InBMap:=InBMap AND DWORD#16#FFFFFFEF;   //CMD_START bit Clear
      END_IF;
END_IF;

// Reset Alarm History Command Response Check
// CONNECT bit & CMD_Start & Setting & CMD_Code
IF (OutBMap AND DWORD#16#00000D91)=16#00000D91 THEN
      RST_ALM_His_Resp := TRUE;
ELSE
      RST_ALM_His_Resp := FALSE;
END_IF;
```

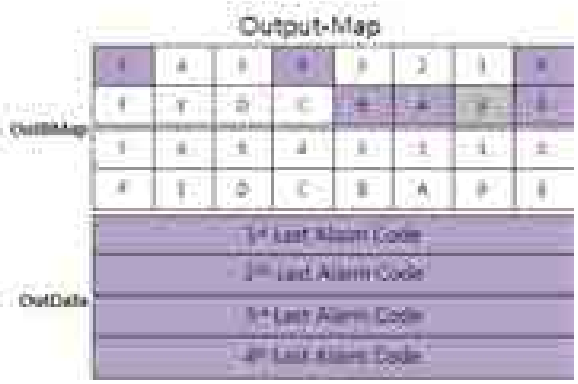
✓ **IO-Map 의 명령 및 응답 형태**

- **알람 내역 초기화 명령 실행**

MOTION/SETTING bit = 1

CMD_CODE = 1110b

CMD_START bit = 1



MOTION/SETTING_RESP bit = 1

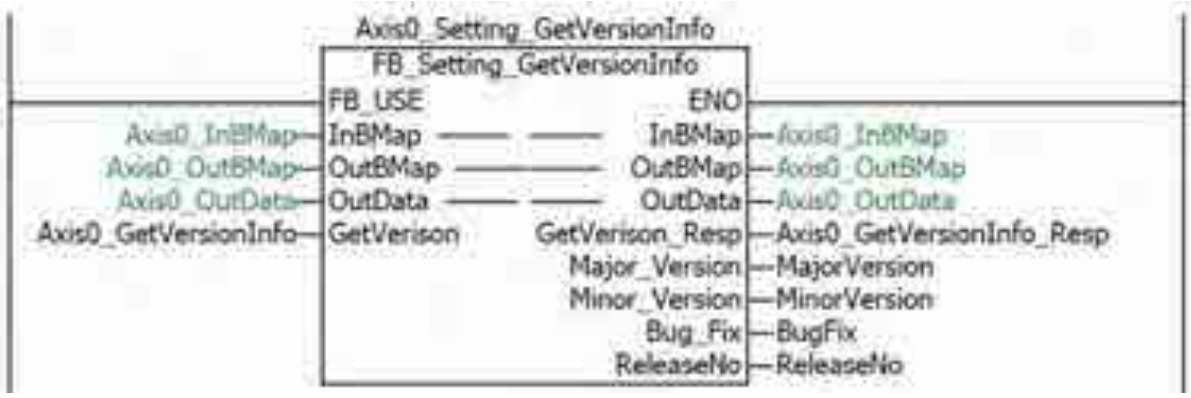
CMD_CODE_RESP = 1110b

CMD_RESP bit = 1

6.9 버전 확인

모션게이트의 버전 확인 명령은 어느 축에서 명령을 실행하여도 모션게이트의 버전 정보를 얻을 수 있습니다. 이 명령으로 설정모드(MOTIONION/SETTING=1)상태의 명령 코드 (CMD_CODE) '5' 에서 동작하여 확인할 수 있습니다.

Function Block 16. 모션게이트 버전 확인



FB_Setting_GetVersionInfo의 사용 변수 목록

Name	In/out	Data Type	Edge	Initial Value	Retain	Constant
InBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutBMap	In/Out	DWORD	No Edge		FALSE	FALSE
OutData	In/Out	DWORD	No Edge		FALSE	FALSE
GetVerison	Input	BOOL	No Edge		FALSE	FALSE
GetVerison_Resp	Output	BOOL	No Edge		FALSE	FALSE
Major_Version	Output	BYTE	No Edge		FALSE	FALSE
Minor_Version	Output	BYTE	No Edge		FALSE	FALSE
Bug_Fix	Output	BYTE	No Edge		FALSE	FALSE
ReleaseNo	Output	BYTE	No Edge		FALSE	FALSE
cmd_check	Internals	BOOL				
VersionInfo	Internals	MG_VERSION_INFO				

Output Data 로 수신되는 데이터는 32 비트 정수로 수신 됩니다. 수신된 데이터는 8 비트 BYTE 형식으로 4 개로 분할됩니다. 이는 열거 형(UNION) 데이터 구조로 나누어 사용할 수 있습니다. 본 매뉴얼에서 사용된 데이터 유형은 다음과 같습니다.

데이터 타입 - Union

Type Name	Member Name	Base Type	Comment
MG_VERSION_INFO		UNION	열거 유형
	Version_Buffer	DWORD	
	VersionNo	ARRAY[0..3] OF BYTE	

ST 33. Get Version Information 명령 처리

```
// Get Version Info Command
IF GetVerison THEN
	// Get Parameter
	InBMap:=(InBMap AND DWORD#16#0000000F) OR DWORD#16#00000590;
	cmd_check := TRUE;
ELSE
	IF cmd_check THEN
	cmd_check := FALSE;
	InBMap:=InBMap AND DWORD#16#FFFFFFEF; //CMD_START bit Clear
	END_IF;
END_IF;

// Get Version Info Command Response Check
// CONNECT bit & CMD_Start & Setting & CMD_Code
IF (OutBMap AND DWORD#16#00000591)=16#00000591 THEN
	VersionInfo.Version_Buffer:= DINT_TO_DWORD(OutData);
	Major_Version := VersionInfo.VersionNo[3];
	Minor_Version:= VersionInfo.VersionNo[2];
	Bug_Fix:= VersionInfo.VersionNo[1];
	ReleaseNo:= VersionInfo.VersionNo[0];
	GetVerison_Resp := TRUE;
ELSE
	GetVerison_Resp := FALSE;
END_IF;
```

✓ **IO-Map 의 명령 및 응답 형태**

- **버전 정보 요청 명령**

MOTION/SETTING bit = 1

CMD_CODE = 0101b

CMD_START bit = 1

MOTION/SETTING_RESP bit = 1

CMD_CODE_RESP = 0101b

CMD_RESP bit = 1

RESPONSE_DATA = Version Information

FASTECH

FASTECH Co., Ltd.

경기도 부천시 원미구 약대동 193번지
부천테크노파크 401동 1202호 (우)420-734
TEL:032)234-6300~1 FAX:032)234-6302
Email : fastech@fastech.co.kr
Homepage : www.fastech.co.kr