

6. 시퀀스 명령어

iM - Σ 시리즈의 내부 시퀀스 프로그램은 컨트롤러 외부의 입출력 장치(S/W 입력, 솔레노이드 On/Off...)의 시퀀스 동작을 제어하기 때문에 별도의 외부 PLC를 사용하지 않아도 제어 시스템 구성이 가능합니다.

그리고, 모든 입출력 접점은 내부의 범용 IO영역 중 아무 곳이나 연결시켜 사용할 수 있으며, 시스템 제어 기능을 처리하기 위한 시스템 접점을 내부의 소프트웨어 접점으로 제공하고 있어서 다양한 운전 기능을 실현할 수 있습니다.

6-1 시퀀스 일반

시퀀스 프로그램은 시스템 접점을 운영하기 위한 **시스템 시퀀스 프로그램**과 일반 입출력 운영을 위한 **사용자 시퀀스 프로그램**을 나누어서 사용할 수 있습니다. **시스템 시퀀스 프로그램**은 시스템 접점에 write 동작을 할 수 있으나 **사용자 시퀀스 프로그램**은 시스템 접점 영역에 write 동작을 할 수 없으므로 H/W 또는 내부 접점을 사용하여 **시스템을 운전(Run, Stop, Org,...)** 하려면 반드시 **시스템 시퀀스 프로그램**을 작성하여 운전하여야 합니다. 컨트롤러에 부착된 전면 패널 스위치로 운전 시에도 마찬가지로 **시스템 시퀀스 프로그램**을 작성하여 운전하여야 합니다.

(매뉴얼 “6-3 시스템 시퀀스 프로그램 예”에 시스템 운전에 대한 예제 프로그램을 참조 하십시오.)

시스템 시퀀스 프로그램은 “SYS.SEQ”라는 파일명만을 사용합니다. **시스템 시퀀스 프로그램**을 사용하기 위해서는 파라미터 **Miscel-Sysseq** 항목을 ‘1’로 설정해야 합니다. 이 항목을 설정하면 컨트롤러에 전원을 투입하여 운전이 개시됨과 동시에 “SYS.SEQ” 프로그램이 실행됩니다. 사용하지 않을 때에는 이 항목을 ‘0’으로 설정합니다. 일단 실행된 **시스템 시퀀스 프로그램**은 전원이 차단될 때까지 멈추지 않습니다.

사용자 시퀀스 프로그램은 다수의 시퀀스 프로그램을 작성할 수 있으며, 그 중 파라미터 **Miscel-Seq Pgm**으로 지정된 프로그램이 선택되어 수행됩니다. 사용자 프로그램은 Operating Loader나 시스템 접점으로 실행시킬 수 있으며, 파라미터 **Miscel-Auto Seq** 항목을 ‘1’로 하여 전원 투입이 되면 자동으로 실행하도록 할 수도 있습니다.

시퀀스 프로그램은 5[ms]마다 1회씩 실행됩니다. (시퀀스 프로그램의 크기가 과도하게 클 경우에는 시스템의 연산 부하 때문에 오동작이 유발될 수 있으므로, 이런 경우에는 프로그램의 크기를 줄여야 합니다.)

6-1-1 입출력 접점 사용

제어기 내부에는 B(0)에서 B(399)까지 총 400 Byte의 접점을 제공하고 있습니다. 각 접점들을 지정할 때에는 'B(23).4'와 같은 형식을 사용합니다.

범용으로 사용할 수 있는 영역은 B(0)~B(255)로, 입출력 카드의 물리 접점을 이 영역에 할당하여 사용할 수 있으며 사용자 시퀀스 프로그램과 시스템 시퀀스 프로그램에서 모두 읽고 쓰기가 가능합니다. 시스템 입력으로 사용할 수 있는 영역은 B(256)~B(299)로, 사용자 시퀀스 프로그램에서는 읽기만 가능하고 시스템 시퀀스 프로그램에서는 읽고 쓰기가 가능합니다. 시스템의 상태를 알려주는 영역은 B(300)~B(319)로, 사용자 시퀀스 프로그램과 시스템 시퀀스 프로그램 모두 읽기만 가능합니다. 그리고 시스템 내부에서 활용하는 영역은 B(320)~B(399)로, 이 영역도 사용자 시퀀스 프로그램과 시스템 시퀀스 프로그램 모두 읽기만 가능합니다.

		User SEQ	System SEQ
B(0)	General Purpose IO (Software IO and Physical IO)	Read / Write	Read / Write
B(255)			
B(256)	System Input		
B(299)			
B(300)	System Status	Read Only	Read Only
B(319)			
B(320)	Internal IO		
B(399)			

6-1-2 범용 입출력 영역

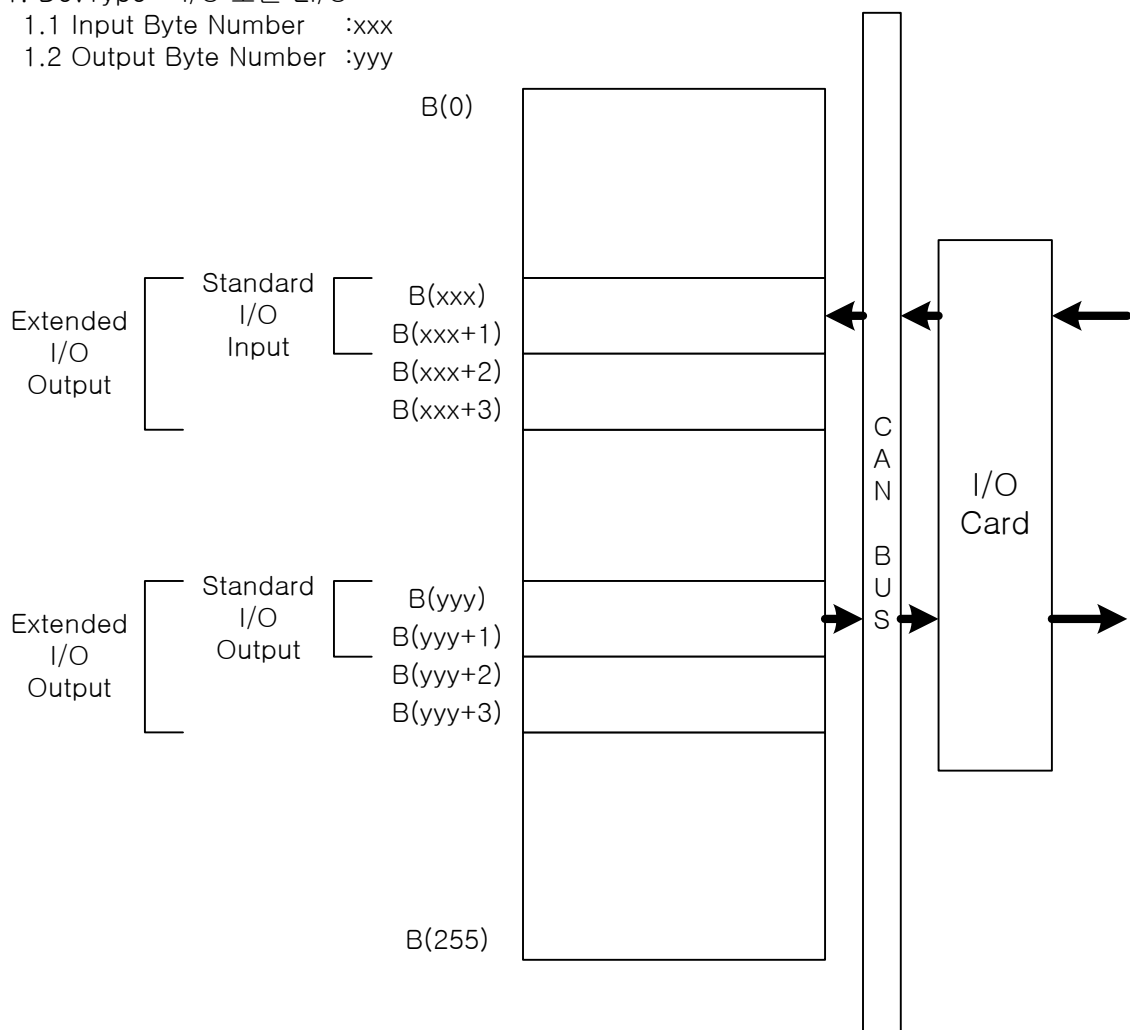
범용 입출력 영역은 소프트웨어 접점으로 사용하거나 입출력 카드의 접점을 할당하여 사용하는 것이 가능합니다. 표준 IO Card는 16/16점(2/2Byte)의 입출력을, 확장 IO Card는 32/32점(4/4Byte)의 입출력을 각각 제공합니다. 파라미터에서 Card의 종류를 선택하고 입출력을 원하는 Byte에 할당하면 시퀀스 프로그램에서 할당된 영역을 물리적인 입출력으로 사용할 수 있습니다.

CAN Parameter

1. DevType = I/O 또는 EI/O

1.1 Input Byte Number :xxx

1.2 Output Byte Number :yyy



6-1-3 시스템 입력 영역

시스템 입력 영역은 제어기의 모든 동작을 통제하는 기능을 갖추고 있습니다. 각 채널의 운전 개시, 원점 이송, jog 운전 등을 명령할 수 있습니다.

B(256)에는 외부의 비상정지, 리셋, 사용자 시퀀스 프로그램의 동작 및 정지 명령이 제공됩니다.

B(260)~B(269)는 채널 1번을 위한 명령이 제공됩니다. B(260)에는 채널의 운전 정지 및 리셋 등의 명령들이 제공됩니다. B(261)은 Jog운전에 대한 보조 명령으로 Jog방식, 방향, 축 선택, 속도 등의 정보를 제공합니다. 이 정보는 Jog운전(JogCmd bit를 '1') 전에 설정해 두어야 하며 Jog중(JogFlag가 '1')에는 변경하면 안됩니다. ProgSel bit는 Run(RunCmd bit를 '1') 하기 전에 운전할 프로그램 번호를 B(263)에 있는 ProgBit로 설정할 때 사용합니다. Run을 '1'로 하기 전에 ProgSel bit와 B(263)정보가 결정되어야 하며, 이 채널의 RunFlag가 '1'이 될 때까지 유지하여야 합니다. MVIOSStart bit는 모션 프로그램의 MVIO 명령과 연계하여 사용하는 것으로 모션 프로그램은 MVIO명령을 받으면 이 bit가 '1'인 것을 확인하고 그 때 B(264)로 지정되는 번호의 위치 데이터를 목표로 하여 이송합니다.

B(270)~B(279), B(280)~B(289), B(290)~B(299)에는 각각 채널 2, 3, 4번을 위한 명령이 제공됩니다. 각 bit의 기능은 B(260)~B(269)와 동일합니다.

	D7	D6	D5	D4	D3	D2	D1	D0	
B(256)	X	X	X	X	USeqStop	USeqRun	Reset	Ext EMG	Common
	X								
B(260)	X	EStopCmd	RstCmd	OriginCmd	JogCmd	StopCmd	StepCmd	RunCmd	Ch1
B(261)	JogSpd1	JogSpd0	JogAxis2	JogAxis1	JogAxis0	JogDir	iJog	JogJoint	
B(262)	THJogData	THJogSpd	MvStbyX	MvStbyT	PgmNoDisp	ProgRst	MVIOStart	ProgSel	
B(263)	ProgBit7	ProgBit6	ProgBit5	ProgBit4	ProgBit3	ProgBit2	ProgBit1	ProgBit0	
B(264)	InputPnt7	InputPnt6	InputPnt5	InputPnt4	InputPnt3	InputPnt2	InputPnt1	InputPnt0	
B(265)	X	SvAllOn	Sv5On	Sv4On	Sv3On	Sv2On	Sv1On	Sv0On	
B(266)	X	SvAllOff	Sv5Off	Sv4Off	Sv3Off	Sv2Off	Sv1Off	Sv0Off	
	X								
B(270)	Ch1의 내용과 동일								Ch2
B(280)	Ch1의 내용과 동일								Ch3
B(290)	Ch1의 내용과 동일								Ch4

6-1-3-1 시스템 입력 영역 상세 내용

1) Common 영역 bit

- ◆ Ext EMG : 외부 비상정지 입력 접점입니다.
‘1’로 Set 되면 전 채널에 비상정지 Alarm 이 발생합니다.
- ◆ Reset : 전 채널 Alarm Reset 기능 접점입니다.
- ◆ UseqRun : 사용자 시퀀스 프로그램 동작 접점입니다.
- ◆ UseqStop : 사용자 시퀀스 프로그램 정지 접점입니다.

2) 각 채널 Command 영역 bit(전 채널 동일한 기능입니다.)

- ◆ RunCmd : 모션 프로그램 운전 접점입니다.
최초 운전 시는 파라미터 MotionPgm에 설정된 프로그램이나 ProgSel 접점으로 선택된 프로그램이 운전 되고 정지 상태시에는 현재의 프로그램을 현재라인부터 실행합니다.
- ◆ StepCmd : 모션 프로그램 스텝 운전 접점입니다.
- ◆ StopCmd : 모션 프로그램 정지 접점입니다.
- ◆ JogCmd : Jog 이동 명령 접점입니다.
Jog 운전에 대한 보조 접점들(JogJoint, iJog, JogDir, JogAxis0~2, JogSpd0~1)과 연계하여 Jog 동작을 수행합니다. iJog시 Edge로 동작합니다.
- ◆ OriginCmd : 원점 수행 접점입니다.
- ◆ RstCmd : 해당 채널 Alarm Reset 기능 접점입니다.
- ◆ EStopCmd : 해당 채널 비상정지 기능 접점입니다.
- ◆ JogJoint : Jog 모션 방법 선택 접점입니다.
‘1’인 경우 Joint 모션(축별 Jog)으로 Jog 이동 합니다.
- ◆ iJog : Jog 이동 방법 선택 접점입니다.
‘1’인 경우 iJog(증분) 이동 합니다.
- ◆ JogDir : Jog 이동의 방향 선택 접점입니다.
‘1’인 경우 CW로 이동합니다.
- ◆ JogAxis0~2 : Jog 이동 할 축 또는 좌표방향 선택 접점입니다.

JogAxis			Joint Jog시 (JogJoint=1)	XY Jog시 (JogJoint=0)
2	1	0		
0	0	0	1번축 이동	X방향 이동
0	0	1	2번축 이동	Y방향 이동
0	1	0	3번축 이동	Z방향 이동
0	1	1	4번축 이동	4번축 이동
1	0	0	5번축 이동	5번축 이동
1	0	1	6번축 이동	6번축 이동
1	1	0	의미 없음	의미 없음
1	1	1	의미 없음	의미 없음

- ◆ JogSpd0~1 : Jog 이동 시 이동 속도 또는 IJog 이동 시 1회 이동 거리를 설정하는 점점입니다. 점점에 따라 변경되는 속도 또는 거리는 파라미터로 변경할 수 있습니다.

JogSpd		사용되는 파라미터
1	0	
0	0	JogSpd0 / JogInch0
0	1	JogSpd1 / JogInch1
1	0	JogSpd2 / JogInch2
1	1	JogSpd3 / JogInch3

- ◆ ProgSel(Edge 동작) : 운전 할 프로그램 선택 확인 점점입니다. ProgSel bit 가 ‘0’ 에서 ‘1’로 변경될 때 ProgBit0~7로 지정되는 프로그램이 선택되어 다음 운전 시 선택된 프로그램이 운전 됩니다. 해당 프로그램이 비어 있는 상태이면 Error 가 발생합니다.
- ◆ MVIOStart(Edge 동작) : 모션 프로그램의 MVIO 명령에 의한 이동 시작 점점입니다. 모션 프로그램에서는 MVIO 명령을 받으면 이 bit가 ON 인 것을 확인하고 그 때 InputPnt0~7로 지정되는 번호의 위치로 이동합니다. 이때 해당 포인트의 데이터가 저장되어 있지 않으면 Run Time 에러가 발생합니다.
- ◆ ProgRst (Edge 동작) : 현재 진행 중이었던 모션 프로그램의 스텝을 초기화 시켜 다음 모션 운전 명령에서 첫 스텝부터 모션 프로그램을 진행하게 하는 점점입니다.
- ◆ PgmNoDisp : 현재 운전 중이거나 운전 대기 중>Loading)인 프로그램 번호를 컨트롤러의 전면 7-Segments 에 표시하는 명령 점점입니다.
- ◆ MvStbyT : TwinX 기구부에서 XYZ 축이 작업 중 일 때 해당 Bit가 On(‘1’) 되면 T 축이 파라미터 Channel - Common - TwinXMov 에서 설정한 대기 위치로 이동합니다.
- ◆ MvStbyX : TwinX 기구부에서 YZT 축이 작업 중 일 때 해당 Bit가 On(‘1’) 되면 X 축이 파라미터 Channel - Common - TwinXMov 에서 설정한 대기 위치로 이동합니다.
- ◆ THJogSpd : 파라미터 Miscel - Option 이 Touch 로 설정되어 있고 해당 Bit 가 On(‘1’) 이되면 터치에서 설정한 JogSpd 값을 사용하여 Jog/iJog 동작을 수행합니다.
- ◆ THJogData: 파라미터 Miscel - Option 이 Touch 로 설정되어 있고 해당 Bit 가 On(‘1’) 이되면 터치에서 설정한 iMoveData 값을 사용하여 iJog 동작을 수행합니다.
- ◆ ProgBit0~7 : 운전할 프로그램 번호 선택 점점입니다.
- ◆ InputPnt0~7 : MVIO 명령어로 이동할 포인트 번호 선택 점점입니다.
- ◆ Sv0On~Sv5On : 축별 서보 On 활성화 선택 점점입니다.

‘1’ 인 경우 축별로 서보 On 이 가능합니다.

◆ SvAllOff : 모든 축을 서보 Off합니다.

◆ Sv0Off~Sv5Off : 축별 서보 Off 활성화 선택 점점입니다.

‘1’ 인 경우 축별로 서보 On 이 가능합니다.

◆ SvAllOff : 모든 축을 서보 Off합니다.

6-1-4 시스템 상태 영역

시스템 상태 영역에서는 제어기 내부의 모든 동작 상태를 확인할 수 있습니다.

B(300)에는 Front Panel의 Key 및 EMG스위치, Operating Loader의 EMG 스위치 상태와 시스템 및 사용자 시퀀스 프로그램의 실행 여부를 확인할 수 있습니다. B(301)에서는 메모리나, CAN bus, 백업상태의 이상과 같은 하드웨어적인 이상 상태 및 시퀀스 프로그램의 실행 불가 상태를 확인할 수 있습니다.

B(304)와 B(305)는 채널 0의 운전 상태를 알려줍니다. 파라미터에 의해 해당 채널을 사용하지 않으면 Active bit는 ‘0’으로 됩니다. Inpos bit는 모션의 이송 동작이 끝난 상태에서 채널에서 사용하는 모든 축의 실제 위치가 파라미터에서 지정하는 InposRang 범위 내로 들어오면 ‘1’이 됩니다.

	D7	D6	D5	D4	D3	D2	D1	D0	
B(300)	FontKey4	FrontKey3	SSeqRunF	USeqRunF	FrontKeyR	FrontKeyL	OP EMG	Front EMG	Common
B(301)	X	X	SSeqErr	USeqErr	Filesys Err	CAN Err	Para Err	Mem Err	
	X								
B(304)	ErrorFlag	OrgOkFlag	JogFlag	OriginFlag	StepFlag	PgmLdFlag	RunFlag	Active	Ch1
B(305)	X	X	X	X	X	X	InRange	InPos	
B(306)	ErrBit7	ErrBit6	ErrBit5	ErrBit4	ErrBit3	ErrBit2	ErrBit1	ErrBit0	
B(307)	X								Ch2
B(308)	Ch1과 동일								
B(312)	Ch1과 동일								Ch3
B(316)	Ch1과 동일								Ch4

6-1-4-1 시스템 상태 영역 상세 내용

1) Common 영역 bit

- ◆ Front EMG : 컨트롤러 전면 패널에 부착된 비상정지 입력을 알리는 점점입니다.
- ◆ OP EMG : Operating Loader 의 비상정지 입력을 알리는 점점입니다.
- ◆ FrontKeyL : 컨트롤러 전면 패널에 부착된 START/ORG SW의 입력을 알리는 점점입니다.
- ◆ FrontKeyR : 컨트롤러 전면 패널에 부착된 STOP/RST SW의 입력을 알리는 점점입니다.
- ◆ USeqRunF : 사용자 시퀀스 프로그램이 운전 중 임을 알리는 점점입니다.
- ◆ SSeqRunF : 시스템 시퀀스 프로그램이 운전 중 임을 알리는 점점입니다.
- ◆ FrontKey3 : DTR의 경우에 본체에 추가로 부착된 ORG SW의 입력을 알리는 점점입니다.
- ◆ FrontKey4 : DTR의 경우에 본체에 추가로 부착된 RST SW의 입력을 알리는 점점입니다.

2) 각 채널 상태 영역 bit

- ◆ Active : 채널이 동작 중으로 설정되었음을 알리는 점점입니다.
파라미터 Channel->Ch1~4 를 None 이 아닌 기구부 형태로 설정하면 '1'로 됩니다.
- ◆ RunFlag : 모션 프로그램이 운전 중 임을 알리는 점점입니다.
- ◆ PgmLdFlag : 모션 프로그램의 컴파일이 성공적으로 수행되었음을 알리는 점점입니다.
- ◆ StepFlag : 모션 프로그램이 스텝 운전 중 임을 알리는 점점입니다.
- ◆ OriginFlag : 원점 수행 중 임을 알리는 점점입니다.
- ◆ JogFlag : Jog 동작 중 임을 알리는 점점입니다.
- ◆ OrgOKFlag : 원점 수행이 완료 되었음을 알리는 점점입니다.
- ◆ ErrorFlag : Error 상태임을 알리는 점점입니다.
- ◆ InPos : 전 축이 파라미터 Channel-Common- InposRang 에서 지정하는 범위로 들어온 상태임을 알리는 점점입니다.
- ◆ InRange : 전 축이 파라미터 Channel-Axis-InRangeL 과 InRangeH 의 범위 내에 들어온 상태임을 알리는 점점입니다.
- ◆ ErrBit0~7 : 현재 발생한 에러의 코드를 출력합니다. 에러가 발생하지 않은 경우 모든 Bit가 0이며, 에러가 발생한 경우 1부터 해당 에러코드가 2진수로 출력됩니다. 에러코드에 대한 상세한 내용은 9장을 참조하십시오.

6-1-5 시스템 내부에서 활용하는 영역

시스템 내부의 활용 영역은 시스템의 동작을 확인하거나 문제점을 추적하는 데에 도움이 됩니다. B(320)~B(336)은 제어기에서 서보로 보내는 명령과 서보의 상태를 알려주는 bit들이 있습니다. 그리고 B(337)~B(344)는 Amp 모듈에 연결되는 각종 센서의 상태를 알려주는 bit들이 있어서 센서의 동작 상태를 검증할 수 있습니다.

	D7	D6	D5	D4	D3	D2	D1	D0		
B(320)	SvOnCmd	AlmRst	BrkOnCmd	Edge2	Edge1	Edge0	ClrCAP	EStop	Servo0~7 Command	Servo Amp Interface
B(321)	Servo 1~7 Command : B(320)과 동일									
~										
B(327)	Servo 1~7 Status : B(328)과 동일								Servo0~7 Status	
B(328)										
B(329)	Servo 1~7 Limit Sensor : B(337)과 동일									
~										
B(336)	ORG	RLS	FLS	X	X	ORG	CW	CCW		
B(337)	Servo 1~7 Limit Sensor : B(337)과 동일									
B(338)									Servo 1~7 Limit Sensor : B(337)과 동일	
~										
B(344)	Servo 1~7 Limit Sensor : B(337)과 동일									
									Servo 1~7 Limit Sensor : B(337)과 동일	

6-1-6 다사 HostPackage 와 인터페이스되는 점점 영역

다사의 iM-Σ 전용 HostPackage 와 연결해서 운전(원점,Jog,Run...)을 하기위한 내부 점점영역 입니다. 제어기 내부에서는 이 영역의 점점이 변할 경우, Host에서 발생한 명령으로 인식하여 해당동작을 수행하게 됩니다.

예를 들어 HostPackage 에 있는 운전 관련 창에서 채널1 의 RUN 버튼을 누르면 B(360).0 이 ON('1') 로 변경됩니다. 제어기의 Background작업에서 이 비트를 검사하여 프로그램을 수행하게 됩니다.

	D7	D6	D5	D4	D3	D2	D1	D0	
B(360)	X	EStopCmd	RstCmd	OriginCmd	JogCmd	StopCmd	StepCmd	RunCmd	Ch1
B(361)	JogSpd1	JogSpd0	JogAxis2	JogAxis1	JogAxis0	JogDir	iJog	JogJoint	
B(362)	X	X	X	X	PgmNoDisp	ProgRst	MVIOStart	ProgSel	
B(363)	ProgBit7	ProgBit6	ProgBit5	ProgBit4	ProgBit3	ProgBit2	ProgBit1	ProgBit0	
B(364)	InputPnt7	InputPnt6	InputPnt5	InputPnt4	InputPnt3	InputPnt2	InputPnt1	InputPnt0	
	X								
B(369)	X	X	X	X	USeqStop	USeqRun	Reset	Ext EMG	Common
B(370)	Ch1의 내용과 동일								Ch2
B(380)	Ch1의 내용과 동일								Ch3
B(390)	Ch1의 내용과 동일								Ch4

6-2 시퀀스 명령어 요약

시퀀스 프로그램의 명령어는 아래와 같이 크게 6가지로 구성되어 있습니다.

구 성	명 령 어	기 능
접점 명령	LOAD	한 회로의 a 접점의 연산 개시
	LOAD NOT	한 회로의 b 접점의 연산 개시
	AND	a 접점의 직렬 접속
	AND NOT	b 접점의 직렬 접속
	OR	a 접점의 병렬 접속
	OR NOT	b 접점의 병렬 접속
결합 명령	AND LOAD	두 블록의 AND 연산 접속
	OR LOAD	두 블록의 OR 연산 접속
반전 명령	NOT	연산 결과의 반전
마스터 컨트롤	MCS	Master Control (Interlock) Set
	MCSC	Master Control (Interlock) Clear
출력 명령	D	입력 조건 Off→On될 때 지정 접점 1Scan On
	D NOT	입력 조건 On→Off 될 때 지정 접점 1Scan On
	SET	접점 출력 On 유지(Set)
	RST	접점 출력 Off 유지(Reset)
	OUT	연산 결과 출력
이동 명령	SR	Shift Data Right
	SL	Shift Data Left
	SC	지정된 접점 self-holding
타이머	TMR	조건에 따라 설정시간만큼 5[ms] 단위로 가산하여 설정시간에 도달 시 출력
카운터	CTR	펄스 입력에 따라 현재치를 +1 증가하여 설정치 이상이 될 때 출력
변환 명령	D2B	BCD 의 값을 2진수로 변환하여 저장
전송 명령	MOVI	전역정수형변수의 값을 다른 전역정수형변수에 복사
증/감 명령	INC	입력이 On 될때마다 해당 Byte 의 값을 1씩 증가
	DEC	입력이 On 될때마다 해당 Byte 의 값을 1씩 감소
종료 명령	PEND	프로그램 종료

6-3 시퀀스 명령어 설명

LOAD

◆ 기능 설명

1. a, b 점점 연산 개시
2. 지정 점점의 On/Off 정보를 결과로 합니다.

◆ 입력 형식

LOAD B(1).0 ; B(1).0 점점을 a점점 연산 개시 합니다.

LOAD NOT B(1).0 ; B(1).0 점점을 b점점 연산 개시 합니다.

OUT

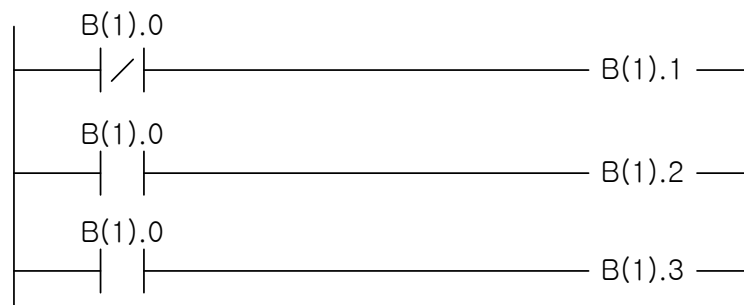
◆ 기능 설명

OUT 명령까지의 연산 결과를 지정한 점점에 출력합니다.X10

◆ 입력 형식

OUT B(1).1 // B(1).1 점점에 출력합니다.

▲ 시퀀스도

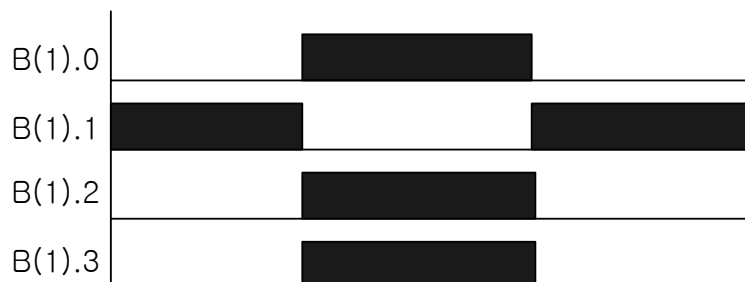


▲ 프로그램 예

```

L005  LOAD NOT B(1).0
L006  OUT B(1).1
L007  LOAD B(1).0
L008  OUT B(1).2
L009  LOAD B(1).0
L010  OUT B(1).3
  
```

▲ 타이밍도



AND

◆ 기능 설명

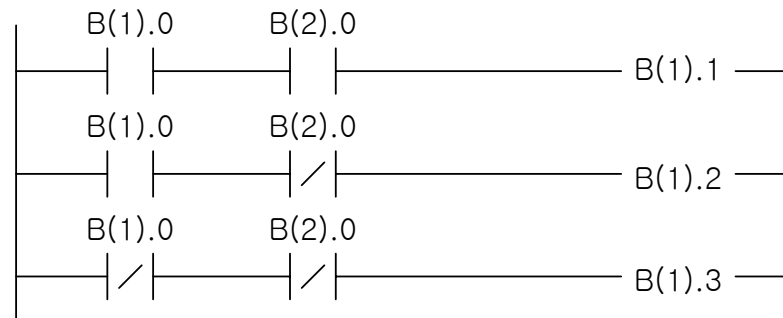
1. a, b 점점의 직렬 접속
2. 지정 점점의 a, b 점점과 직렬로 연결된 점점을 AND 연산하여 이것을 결과로 합니다.

◆ 입력 형식

AND B(1).0 // B(1).0 점점을 a점점 직렬 접속 합니다.

AND NOT B(2).0 // B(2).0 점점을 b점점 직렬 접속 합니다.

▲ 시퀀스도



▲ 프로그램 예

```

L005  LOAD B(1).0
L006  AND B(2).0
L007  OUT B(1).1
L008  LOAD B(1).0
L009  AND NOT B(2).0
L010  OUT B(1).2
L011  LOAD NOT B(1).0
L012  AND NOT B(2).0
L013  OUT B(1).3

```

▲ 타이밍도



OR

◆ 기능 설명

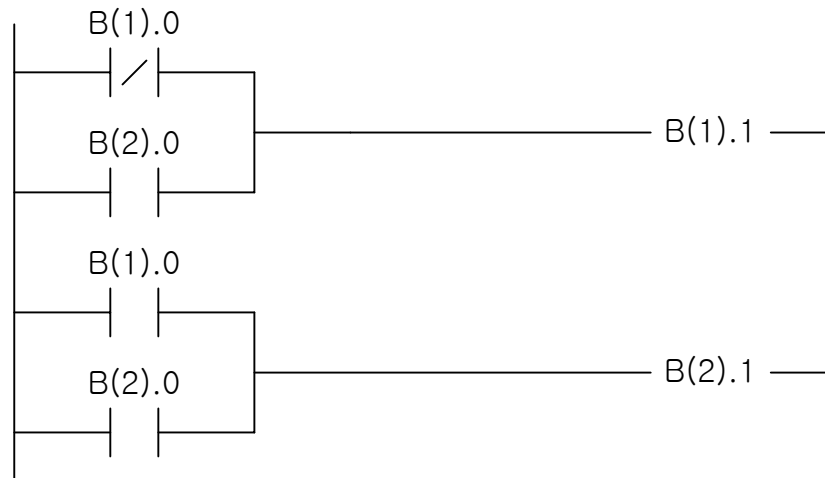
1. a, b 점점의 병렬 접속
2. 지정 점점의 a, b 점점과 병렬로 연결된 점점을 OR 연산하여 이것을 결과로 합니다.

◆ 입력 형식

OR B(1).0 // B(1).0 점점을 a점점 병렬 접속 합니다.

OR NOT B(2).0 // B(2).0 점점을 b점점 병렬 접속 합니다.

▲ 시퀀스도

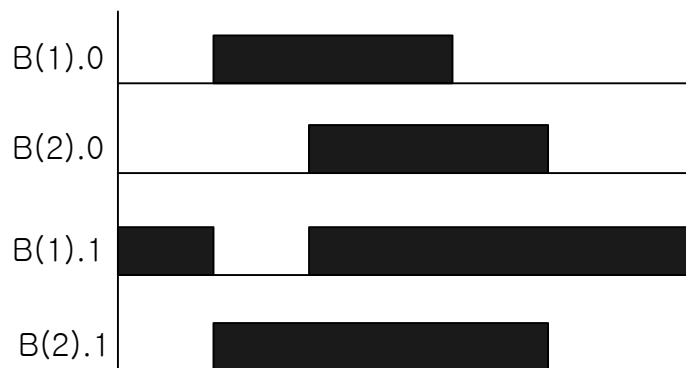


▲ 프로그램 예

```

L005  LOAD NOT B(1).0
L006  OR B(2).0
L007  OUT B(1).1
L008  LOAD B(1).0
L009  OR B(2).0
L010  OUT B(2).1
    
```

▲ 타이밍도



AND LOAD

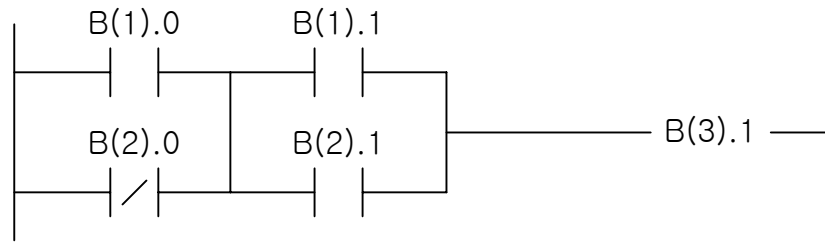
◆ 기능 설명

두 블록의 직렬(AND) 연산 접속

◆ 입력 형식

AND LOAD

▲ 시퀀스도



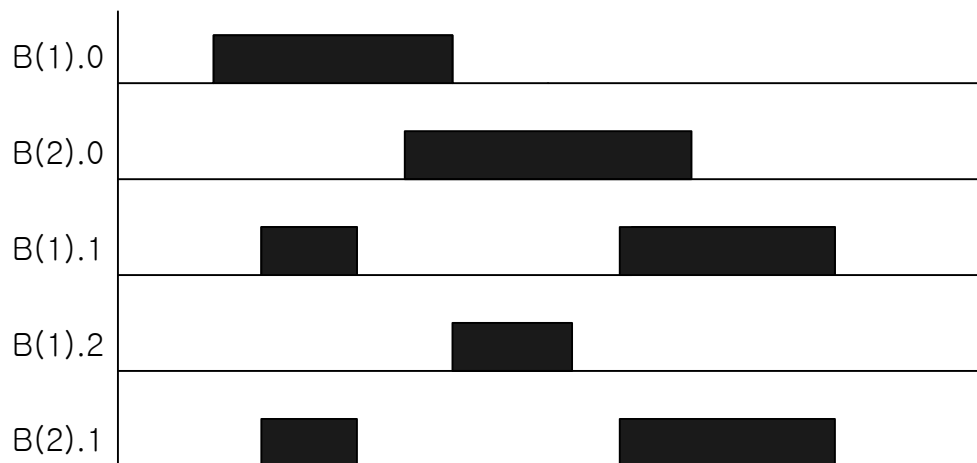
▲ 프로그램 예

```

L005  LOAD B(1).0
L006  OR NOT B(2).0
L007  LOAD B(1).1
L008  OR B(2).1
L009  AND LOAD
L010  OUT B(3).1

```

▲ 타이밍도



OR LOAD

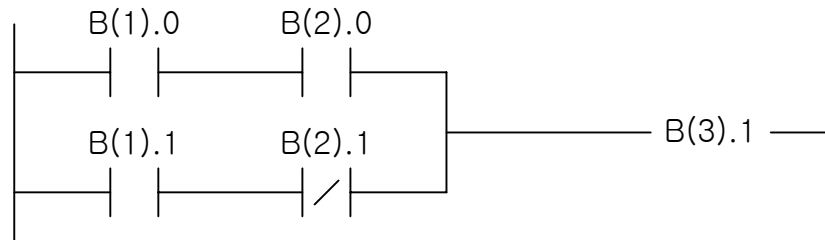
◆ 기능 설명

두 블록의 병렬(OR) 연산 접속

◆ 입력 형식

OR LOAD

▲ 시퀀스도

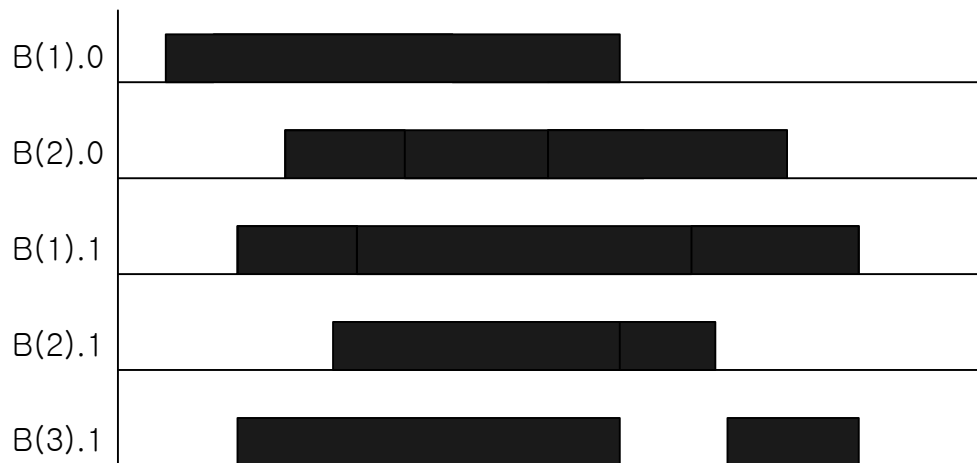


▲ 프로그램 예

```

L005  LOAD B(1).0
L006  AND B(2).0
L007  LOAD B(1).1
L008  AND NOT B(2).1
L009  OR LOAD
L010  OUT B(3).1
    
```

▲ 타이밍도



NOT

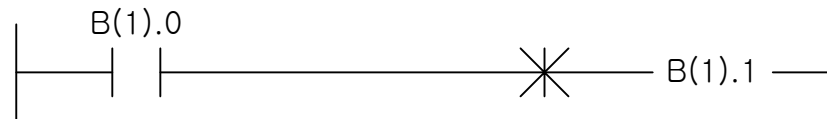
◆ 기능 설명

1. 연산 결과의 반전
2. 연산 결과의 반전은 다음과 같이 됩니다.
 - 1) a 점점 → b 점점
 - 2) b 점점 → a 점점
 - 3) 직렬 연결 → 병렬 연결
 - 4) 병렬 연결 → 직렬 연결

◆ 입력 형식

NOT

▲ 시퀀스도



▲ 프로그램 예

```

L005  LOAD NOT B(1).0
L006  OUT B(1).1
  
```

▲ 타이밍도



D

◆ 기능 설명

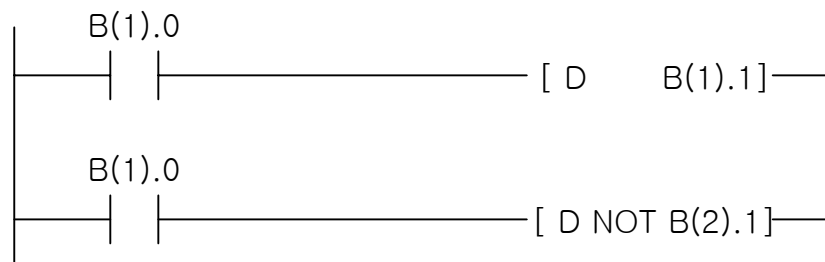
1. 입력 조건 On ↔ Off될 때 지정 점점 1Scan On
2. 한 프로그램에서 사용 가능 횟수 : 288회

◆ 입력 형식

D B(1).1 // 입력 조건이 Off → On될 때 B(1).1 점점이 1 Scan On되며
이외에는 Off됩니다.

D NOT B(2).1 // 입력 조건이 On → Off될 때 B(2).1 점점이 1 Scan On되며
이외에는 Off됩니다.

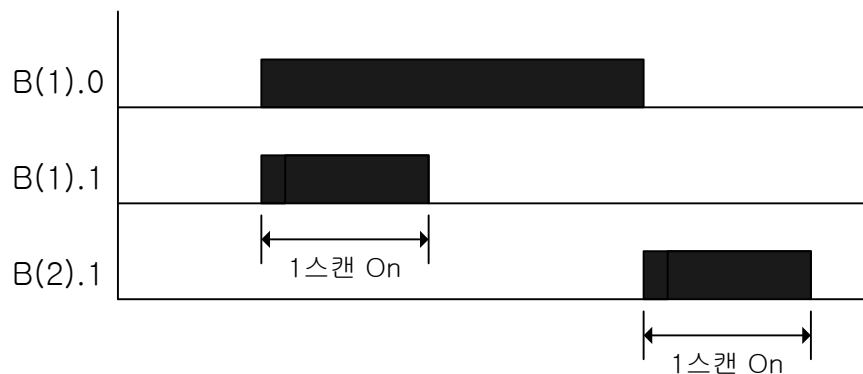
▲ 시퀀스도



▲ 프로그램 예

```
L005  LOAD B(1).0
L006  D B(1).1
L007  OUT B(1).0
L008  D NOT B(2).1
```

▲ 타이밍도



SET

◆ 기능 설명

1. 점점 출력 On 유지(Set).
2. 입력 조건이 On되면 지정 점점을 On상태로 유지시켜 입력 조건이 Off되어도 지정 점점은 계속 On이 됩니다.

◆ 입력 형식

SET B(1).1 // B(1).1 점점을 On 상태로 유지합니다.

RST

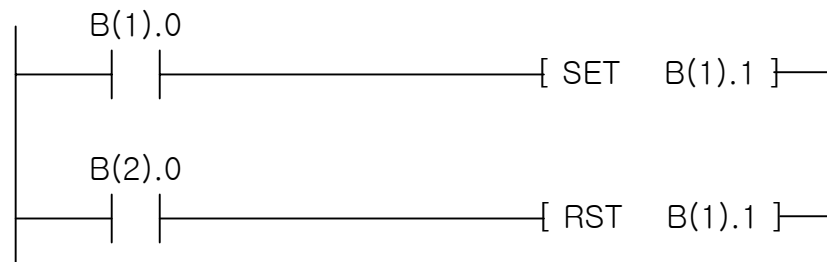
◆ 기능 설명

1. 점점 출력 Off 유지(Reset).
2. 입력 조건이 On되면 지정 점점을 Off상태로 유지시켜 입력 조건이 Off되어도 지정 점점은 계속 Off가 됩니다.

◆ 입력 형식

RST B(1).1 // B(1).1 점점을 Off 상태로 유지합니다.

▲ 시퀀스도



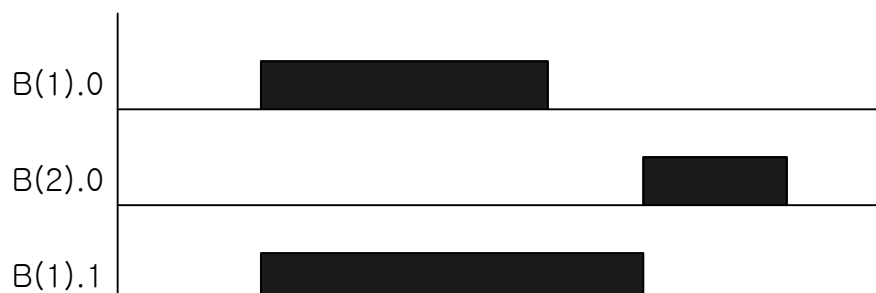
▲ 프로그램 예

```

L005  LOAD B(1).0
L006  SET B(1).1
L007  LOAD B(2).0
L008  RST B(1).1

```

▲ 타이밍도



MCS / MCSC

◆ 기능 설명

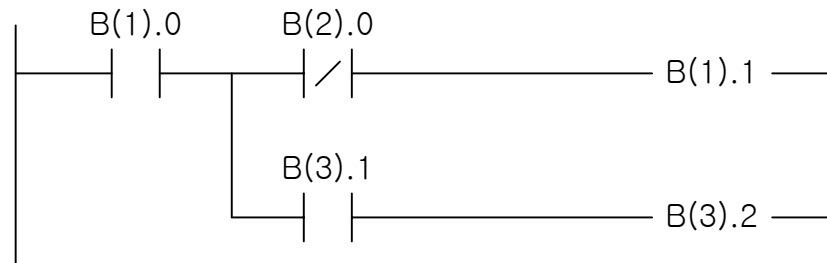
1. 인터록 Set / Clear
2. MCS의 입력 조건이 On이 되면 MCS ~ MCSC 까지를 실행하고 Off되면 실행하지 않습니다.
3. 우선 순위는 먼저 실행한 것이 우선 순위가 되고 해제는 역순으로 합니다.
4. 우선 순위가 높은 것이 해제되면 낮은 순위의 MCS~MCSC 블록도 해제됩니다.

◆ 입력 형식

MCS // 인터록을 Set합니다.

MCSC // 인터록을 Clear합니다.

▲ 시퀀스도



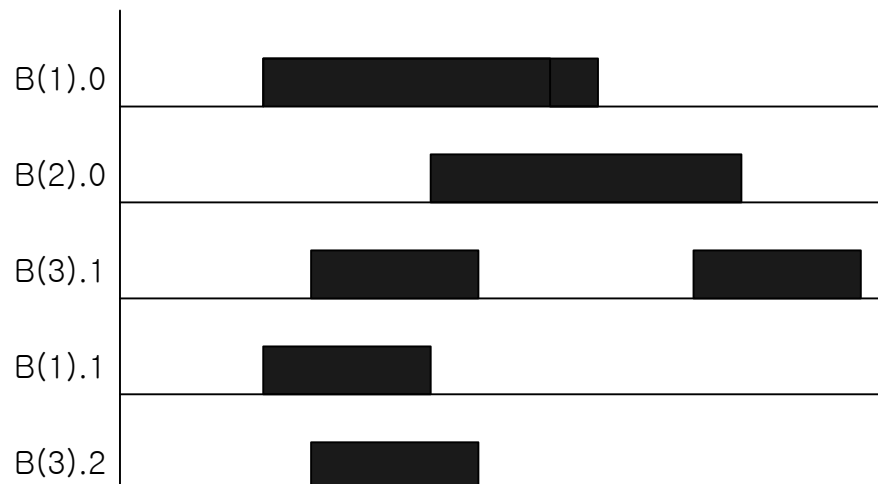
▲ 프로그램 예

```

L005  LOAD B(1).0
L006  MCS
L007  LOAD NOT B(2).0
L008  OUT B(1).1
L009  LOAD B(3).1
L010  OUT B(3).2
L011  MCSC

```

▲ 타이밍도



SR

◆ 기능 설명

1. Shift Data Right
2. Shift Data와 Shift Pulse의 두개의 입력을 갖습니다.
3. Shift Pulse에 의해서 Data는 1 bit씩 오른쪽으로 이동하며 최상위 bit(MSB)에는 Shift 값이 들어가고 최하위 bit(LSB)에 있던 값은 소멸합니다.

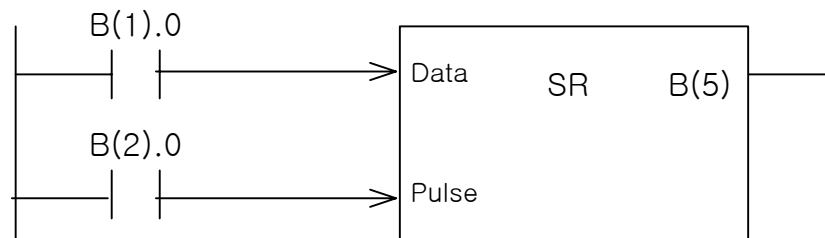
◆ 입력 형식

LOAD B(1).0 ; B(1).0을 Shift Data로 설정합니다.

LOAD B(2).0 ; B(1).0을 Shift Pulse로 설정합니다.

SR B(5) ; B(5)의 값을 Shift Data와 Shift Pulse에 의해서 오른쪽으로 1bit씩 이동합니다.

▲ 시퀀스도

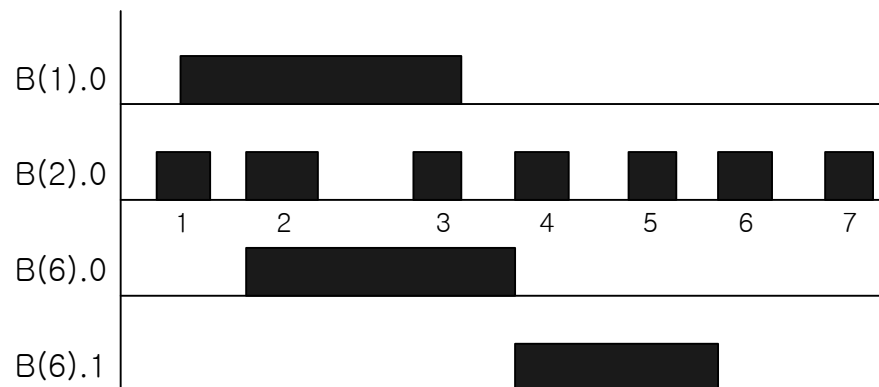


▲ 프로그램 예

```

L005  LOAD B(1).0
L006  LOAD B(2).0
L007  SR B(5)
L008  LOAD B(5).7
L009  OUT B(6).0
L008  LOAD B(5).5
L009  OUT B(6).1
  
```

▲ 타이밍도



SL

◆ 기능 설명

1. Shift Data Left
2. Shift Data와 Shift Pulse의 두개의 입력을 갖습니다.
3. Shift Pulse에 의해서 Data는 1 bit씩 왼쪽으로 이동하며 최하위 bit(LSB)에 는 Shift 값이 들어가고 최상위 bit(MSB)에 있던 값은 소멸합니다.

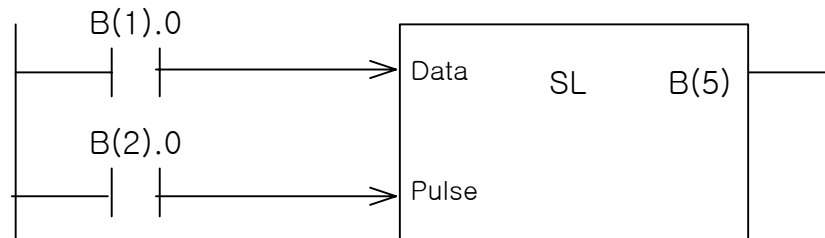
◆ 입력 형식

LOAD B(1).0 // B(1).0을 Shift Data로 설정합니다.

LOAD B(2).0 // B(1).0을 Shift Pulse로 설정합니다.

SL B(5) // B(5)의 값을 Shift Data와 Shift Pulse에 의해서 왼쪽으로 1bit씩 이동합니다.

▲ 시퀀스도

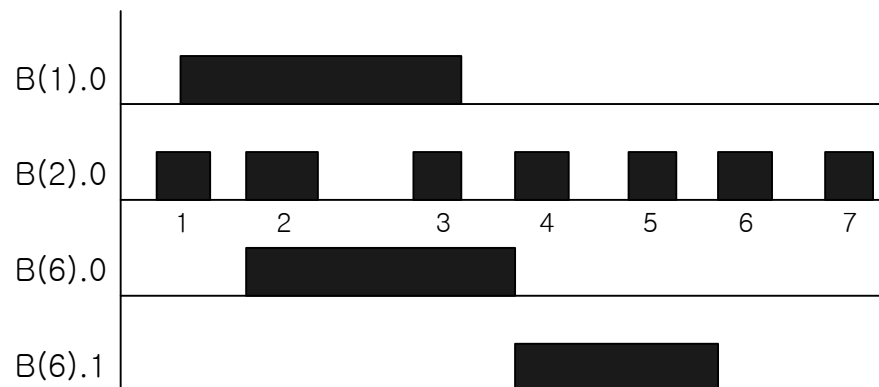


▲ 프로그램 예

```

L005  LOAD B(1).0
L006  LOAD B(2).0
L007  SL B(5)
L008  LOAD B(5).0
L009  OUT B(6).0
L008  LOAD B(5).2
L009  OUT B(6).1
    
```

▲ 타이밍도



SC

◆ 기능 설명

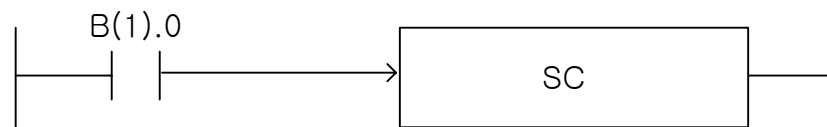
1. 지정된 점점 Self-Holding
2. 입력이 ON되면 해당 출력을 Self-Holding 시켜 입력이 Off 되어도 출력을 유지 시킵니다.
3. 이 때 해당 Byte의 다른 점점들은 자동적으로 Off 됩니다.

◆ 입력 형식

LOAD B(1).0 // B(1).0을 Loading 합니다.

SC B(5).0 // B(5).0의 값을 B(1).0 이 ON되면 Self-Holding 합니다.

▲ 시퀀스도



▲ 프로그램 예

```

L005  LOAD B(1).0
L006  SC B(5).0
L007  LOAD B(1).1
L008  SC B(5).1
  
```

▲ 타이밍도



TMR()

◆ 기능 설명

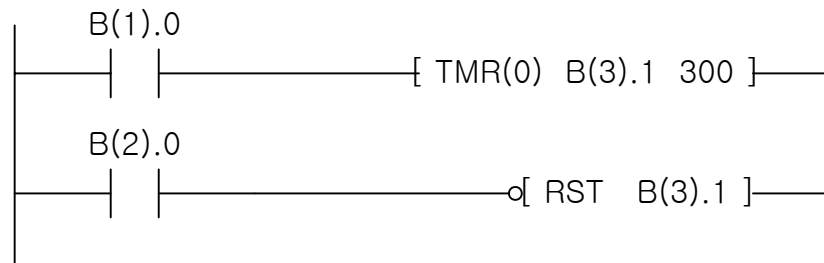
1. 조건에 따라 설정 시간만큼 가산하여 설정시간 될 때 출력
2. Data 입력 범위 : 1 ~ 65536
3. 가산 시간 : 5[ms]
4. 최대 사용 TMR() 개수 : TMR(0) ~ TMR(63) 총 64개

◆ 입력 형식

TMR(0) B(3).1 <D> 300 ; B(3).1 접점을 300x5[ms] 후에 출력합니다.

TMR(0) B(3).1 <D> GINT(0) ; B(3).1 접점을 전역 정수형 변수 GINT(0)에 설정된 값 X 5[ms] 후에 출력합니다.

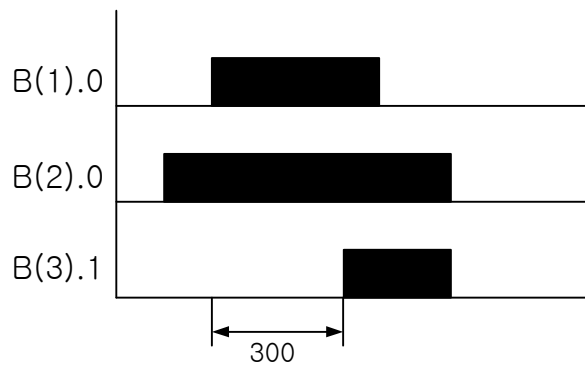
▲ 시퀀스도



▲ 프로그램 예

```
L005  LOAD B(2).0           // Reset
L006  LOAD B(1).0           // Timer 시작
L007  TMR(0) B(3).1 <D> 300
```

▲ 타이밍도



CTR()

◆ 기능 설명

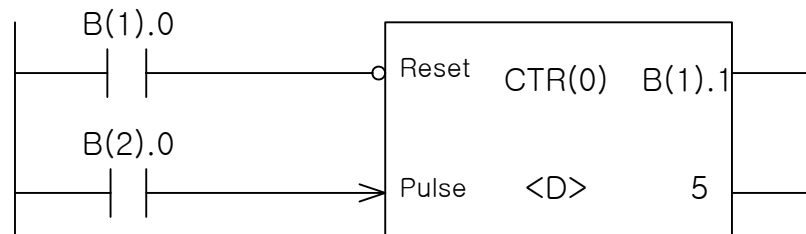
1. 펄스 입력에 따라 현재치를 +1 증가하여 설정치 이상이 될 때 출력
2. Reset 신호가 입력(On→Off)되면 현재치는 “0”이 됩니다.
3. Data 입력 범위 : 1 ~ 65536
4. 최대 사용 CTR() 개수 : CTR(0) ~ CTR(63) 총 64개

◆ 입력 형식

CTR(0) B(1).1 <D> 5 ; B(1).1 접점을 설정 접점이 5회 입력되면 출력합니다.

CTR(0) B(1).1 <D> GINT(0) ; B(1).1 접점을 설정 접점이 전역 정수형 변수 GIN(0)에 설정된 횟수 만큼 입력되면 출력합니다.

▲ 시퀀스도



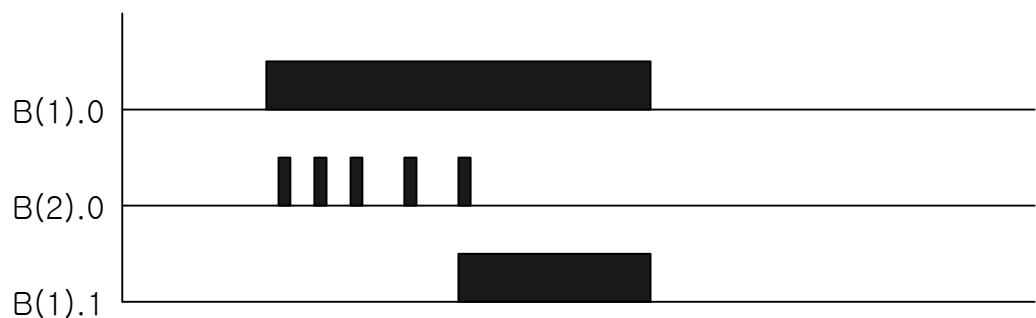
▲ 프로그램 예

```

L005  LOAD B(1).0          // Reset
L006  LOAD B(2).0          // Count Pulse
L007  CTR(0) B(1).1 <D> 5

```

▲ 타이밍도



D2B

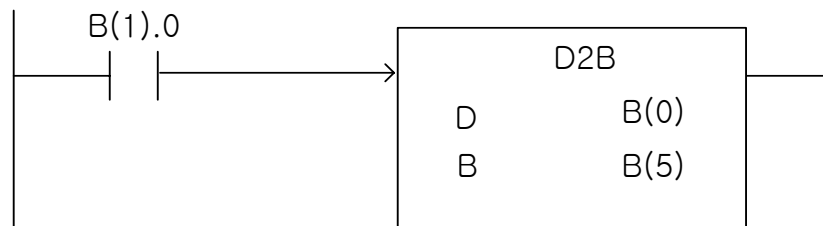
◆ 기능 설명

1. 펄스 입력에 따라 해당 Byte 의 BCD 값을 2진수로 변경하여 저장합니다.

◆ 입력 형식

D2B B(0) B(5) ; 입력이 On 될 때마다 B(0) BCD 값을 2진수로 변경하여 B(5)에 저장합니다.

▲ 시퀀스도



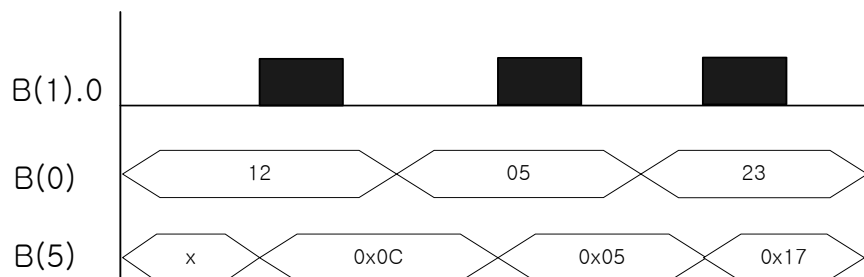
▲ 프로그램 예

```

L005  LOAD B(1).0
L006  D2B B(0) B(5)

```

▲ 타이밍도



MOVI

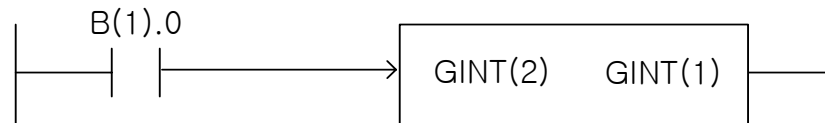
◆ 기능 설명

1. 펄스 입력에 따라 전역 정수형 변수의 값을 해당 전역 정수형 변수에 복사

◆ 입력 형식

MOVI GINT(2) GINT(1) ; 입력이 On 될 때마다 GINT(2)의 값을 GINT(1)에 복사합니다..

▲ 시퀀스도

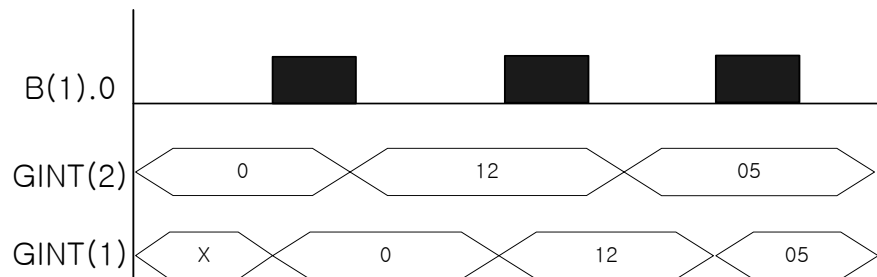


▲ 프로그램 예

```

L005  LOAD B(1).0
L006  MOVI GINT(2) GINT(1)
  
```

▲ 타이밍도



INC

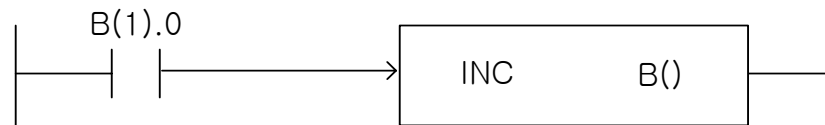
◆ 기능 설명

2. 펄스 입력에 따라 해당 Byte 의 값을 1씩 증가

◆ 입력 형식

INC B(5) ; 입력이 On 될 때마다 B(5)의 값을 1씩 증가합니다.

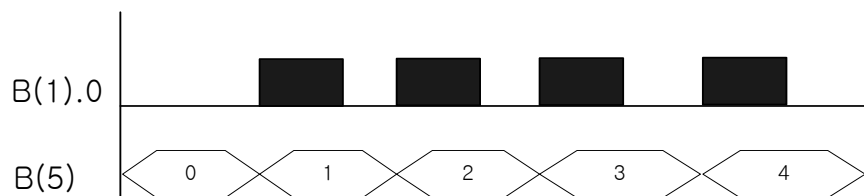
▲ 시퀀스도



▲ 프로그램 예

```
L005  LOAD B(1).0
L006  INC B(5)
```

▲ 타이밍도



DEC

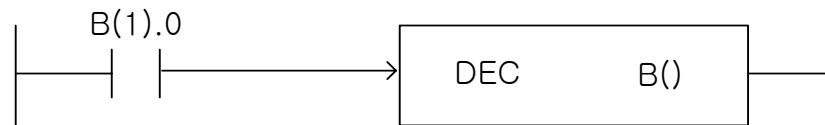
◆ 기능 설명

1. 펄스 입력에 따라 해당 Byte 의 값을 1씩 감소

◆ 입력 형식

INC B(5) ; 입력이 On 될 때마다 B(5)의 값을 1씩 감소합니다.

▲ 시퀀스도

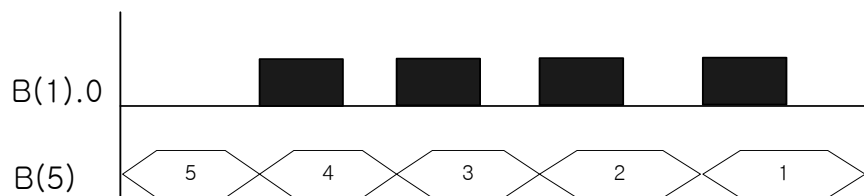


▲ 프로그램 예

```

L005  LOAD B(1).0
L006  DEC B(5)
  
```

▲ 타이밍도



PEND

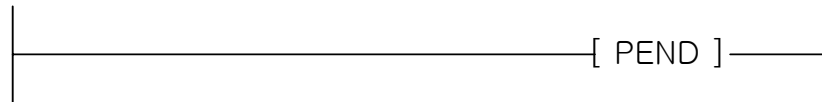
◆ 기능 설명

1. 프로그램을 종료합니다.
2. 시퀀스 프로그램 스캔(Scan) 중 PEND 명령을 만나면 L000 Line으로 돌아가 다시 스캔을 시작합니다.
3. PEND 명령은 반드시 프로그램의 마지막에 입력해야 합니다.

◆ 입력 형식

PEND ; 부 명령어 없이 단독으로 사용합니다.

▲ 시퀀스도



▲ 프로그램 예

```
L000  LOAD B(1).0
      :
L150  PEND
```

6-3 시스템 시퀀스 프로그램 예

6-3-1 전 채널 동시 운전의 경우

프로그램 설명

1. 4개의 채널을 사용한다.
2. H/W Input B(0) Port에 입력되는 신호로 다음의 기능을 구현한다.
 - ◆ B(0).0 : 전 채널 RUN 입력
 - ◆ B(0).1 : 전 채널 STOP 입력
 - ◆ B(0).2 : 전 채널 Origin 입력
 - ◆ B(0).3 : 전 채널 EMG 입력
 - ◆ B(0).4 : 전 채널 Alarm Reset 입력

프로그램 예

```
// RUN
LOAD B(0).0
AND B(304).0          // 채널1 Active 조건과 AND
D B(260).0            // 채널1 Run Command
LOAD B(0).0
AND B(308).0          // 채널2 Active 조건과 AND
D B(270).0            // 채널2 Run Command
LOAD B(0).0
AND B(312).0          // 채널3 Active 조건과 AND
D B(280).0            // 채널3 Run Command
LOAD B(0).0
AND B(316).0          // 채널3 Active 조건과 AND
D B(290).0            // 채널3 Run Command
// STOP
LOAD B(0).1
D B(260).2            // 채널1 Stop Command
LOAD B(0).1
D B(270).2            // 채널2 Stop Command
LOAD B(0).1
D B(280).2            // 채널3 Stop Command
LOAD B(0).1
D B(290).2            // 채널4 Stop Command
// ORG
LOAD B(0).2
AND B(304).0          // 채널1 Active 조건과 AND
D B(260).4            // 채널1 Origin Command
```

```
LOAD B(0).2
AND B(308).0      // 채널2 Active 조건과 AND
D B(270).4        // 채널2 Origin Command
LOAD B(0).2
AND B(312).0      // 채널3 Active 조건과 AND
D B(280).4        // 채널3 Origin Command
LOAD B(0).2
AND B(316).0      // 채널4 Active 조건과 AND
D B(290).4        // 채널4 Origin Command
// ESTOP
LOAD B(0).3
D B(256).0
// Reset
LOAD B(0).4
D B(256).1
PEND
```

6-3-2 채널 간 원점 순서가 다른 운전의 경우

프로그램 설명

1. 3개의 채널을 사용하고 채널2,3 의 원점 복귀가 완료 된 후 채널1의 원점 복귀를 수행한다.
 2. H/W Input B(0).4 Port에 입력되는 신호를 원점 복귀 신호로 사용한다.
- ◆ B(0).4 : 사용자 원점 복귀 신호

프로그램 예

```

LOAD B(0).4           // 사용자 원점 복귀 입력
AND NOT B(304).1       // 채널 1 Run Flag(운전 중이 아니면)
AND NOT B(308).1       // 채널 2 Run Flag(운전 중이 아니면)
AND NOT B(312).1       // 채널 3 Run Flag(운전 중이 아니면)
D B(270).4             // 채널 2 Org Command
LOAD B(270).4          // 채널 2 Org Command
D B(280).4             // 채널 3 Org Command
LOAD B(308).4          // 채널 2 Org 중 Flag
AND B(312).4           // 채널 3 Org 중 Flag
SET B(100).0           // 내부 변수(채널 2,3 이 원점 동작을 시작 했음)
LOAD B(100).0
AND NOT B(308).4       // 채널 2 Org 중 Flag
AND NOT B(312).4       // 채널 3 Org 중 Flag
SET B(100).1           // 내부 변수(채널 2,3 이 원점 동작을 완료 했음)
LOAD B(100).1
RST B(100).0
LOAD B(100).1          // 채널 2,3 Org 완료 후
D B(260).4             // 채널 1 Org Command
LOAD B(100).1          // 채널 1 Org Command
AND B(304).4           // 채널 1 Org 중 Flag
SET B(100).2           // 내부 변수(채널 1 이 원점 동작을 시작 했음)
LOAD B(100).2
RST B(100).1
LOAD B(100).2
AND NOT B(304).4       // 채널 1 Org 중 Flag(채널 1 이 원점 동작을 완료 했음)
RST B(100).2
PEND

```

6-3-3 컨트롤러 전면 패널 스위치를 이용한 운전의 경우

프로그램 설명

1. 컨트롤러 전면 패널에 부착되어 있는 2개의 스위치를 이용하여 다음 기능의 운전을 한다.
2. 2개의 스위치 : START/ORG, STOP/RST
3. 운전 기능
 - RUN, STOP, ORG, RESET, 프로그램 Reset
4. 스위치의 기능
 - START/ORG : 원점 미 수행 전에는 원점 복귀 입력으로 동작하고 원점 완료 후에는 프로그램 운전으로 동작한다.
 - STOP/RST : 운전 중에는 정지, 알람 상태에서는 Reset, 정지 상태에서는 0.5초간 누르고 있으면 프로그램 Reset(초기화)으로 동작한다.

프로그램 예

```

LOAD B(300).2          // Run/Org Panel S/W Load
AND B(304).0            // 채널 1 Active Flag
AND NOT B(304).6        // 채널 1 Org OK Flag
AND NOT B(304).1        // 채널 1 Run Flag
D B(260).4              // 채널 1 Org Command
LOAD B(300).2          // Run/Org Panel S/W Load
AND B(304).0            // 채널 1 Active Flag
AND NOT B(304).4        // 채널 1 Org 중 Flag
D B(260).0              // 채널 1 Run Command
LOAD B(300).3           // Stop/Reset Panel S/W Load
AND B(304).1            // 채널 1 Run 중 Flag
D B(260).2              // 채널 1 Stop Command
LOAD B(300).3           // Stop/Reset Panel S/W Load
AND B(304).7            // 채널 1 Error Flag
D B(260).5              // 채널 1 Reset Command
LOAD B(300).3           // Stop/Reset Panel S/W Load
AND NOT B(304).7        // 채널 1 Error Flag(정상상태)
AND NOT B(304).1        // 채널 1 Run 중 Flag(Stop 상태)
D B(100).0              // Alarm 이 아닌 정지 상태에서 Reset S/W 눌림
LOAD B(100).0
LOAD B(100).0
TMR(0) B(101).0 <D> 100 // 0.5s Timer
LOAD B(101).0
D B(262).2              // 채널 1 Pgm Reset
LOAD NOT B(304).2       // 채널 1 PgmLoad Flag(프로그램이 초기화 되었으면)
    
```

```
RST B(262).2      // Pgm Reset Clear  
LOAD NOT B(304).2  
RST B(100).0  
LOAD NOT B(304).2  
RST B(101).0  
PEND
```