참조 매뉴얼

Allen-Bradley

Logix 5000 Controller 일반 명령 참조 매뉴얼

1756 ControlLogix, 1756 GuardLogix, 1769 CompactLogix, 1769 Compact GuardLogix, 1789 SoftLogix, 5069 CompactLogix, Emulate 5570

Allen-Bradley • Rockwell Software

Rockwell Automation

중요한 사용자 정보

이 제품을 설치, 구성, 작동 또는 유지보수하기 전에 이 문서와 이 장비의 설치, 구성 및 작동에 대한 추가 리소스 섹션에 표시된 문서를 읽으십시오. 사용자는 적용 가능한 모든 규정, 법률, 표준의 요구 사항 외에도 설치 및 연결 지침을 숙지해야 합니다.

설치, 조정, 가동, 사용, 조립, 분해 및 유지보수를 포함한 활동은 적절한 교육을 받은 사람이 해당 직업 규약에 따라 수행해야 합니다. 제조업체에서 지정하지 않은 방식으로 이 장비를 사용할 경우 장비에서 제공하는 보호가 손상될 수 있습니다.

Rockwell Automation, Inc.는 어떠한 경우에도 이 장비의 사용 또는 적용으로 인해 발생한 간접 또는 결과적 손해에 대해 책임을 지지 않습니다.

이 설명서의 예와 다이어그램은 설명용으로만 제공됩니다. 특정 설치와 관련된 변수와 요구 사항이 많으므로 Rockwell Automation, Inc.는 그러한 예와 다이어그램을 기반으로 한 실제 사용에 대해 책임을 지지 않습니다.

Rockwell Automation, Inc.는 이 설명서에 명시된 정보, 회로, 장비 또는 소프트웨어의 사용과 관련한 특허 책임을 지지 않습니다.

Rockwell Automation, Inc.의 서면 허가 없이 이 설명서의 내용을 전부 또는 일부 복제하는 행위는 금지됩니다.

필요한 경우 이 설명서 전체에서 주석을 사용해 사용자에게 안전 고려사항을 전달합니다.

경고: 위험한 환경에서 부상, 사망, 재산 피해 또는 경제적 손실로 이어질 수 있는 폭발을 유발할 수 있는 관행 또는 상황에 대한 정보를 나타냅니다.

주의: 부상, 사망, 재산 피해 또는 경제적 손실로 이어질 수 있는 관행 또는 상황에 대한 정보를 나타냅니다. 주의는 위험을 식별하고, 위험을 피하고, 결과를 인지하는 데 도움이 됩니다.

중요: 제품을 성공적으로 적용하고 이해하는 데 필요한 정보를 나타냅니다.

특정 주의사항을 전달하기 위해 장비의 위 또는 내부에 라벨을 부착할 수도 있습니다.

감전 위험: 사용자에게 위험 전압이 있을 수 있음을 경고하기 위해 장비(예: 드라이브 또는 모터)의 위 또는 내부에 라벨을 부착할 수 있습니다.

화상 위험: 사용자에게 표면이 위험 온도에 도달할 수 있음을 경고하기 위해 장비(예: 드라이브 또는 모터)의 위 또는 내부에 라벨을 부착할 수 있습니다.

아크플래시 위험: 사용자에게 잠재적인 아크플래시를 경고하기 위해 장비(예: 모터 제어 센터)의 위 또는 내부에 라벨을 부착할 수 있습니다. 아크플래시는 심각한 부상 또는 사망을 유발합니다. 적절한 개인 보호 용구(PPE)를 착용하십시오. 안전한 작업 관행과 개인 보호 용구(PPE)에 대한 모든 규제 요구 사항을 따르십시오.

이 설명서에는 새로운 정보와 업데이트된 정보가 포함되어 있습니다. 다음 참조 표를 사용하여 변경된 정보를 찾을 수 있습니다.

글로벌 변경사항

이번 릴리스에는 없습니다.

새로운 기능 또는 향상된 기능

이 표에는 이 버전에서 변경된 항목의 목록, 변경 이유, 변경된 정보가 포함된 항목에 대한 링크가 나와 있습니다.

항목 이름	이유
알람 설정 동작(ASO) 페이지의 76	새 알람 명령어
알람 명령어 페이지의 29	알람 설정 동작(ASO) 명령어가 항목에 추가되었습니다.
닫힘 검사(XIC) 페이지의 82	새 데이터 유형이 추가되었습니다.
열림 검사(XIO) 페이지의 84	새 데이터 유형이 추가되었습니다.
출력 전원 공급(OTE) 페이지의 103	새 데이터 유형이 추가되었습니다.
출력 래치(OTL) 페이지의 105	새 데이터 유형이 추가되었습니다.
출력 래치 해제(OTU) 페이지의 108	새 데이터 유형이 추가되었습니다.
비교 명령어 페이지의 329	펑션 블록 다이어그램 기능의 새 그래픽 이미지가 추가되었습니다.
같음(EQU) 페이지의 334	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
큼(GRT) 페이지의 343	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
크거나 같음(GEQ) 페이지의 352	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.

항목 이름	이유
작음(LES) 페이지의 361	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
작거나 같음(LEQ) 페이지의 370	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
제한(LIM) 페이지의 379	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
마스크 같음(MEQ) 페이지의 389	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
같지 않음(펑션 NEQ) 페이지의 398	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
절대값(ABS) 페이지의 412	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
더하기(ADD) 페이지의 419	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
계산(CPT) 페이지의 425	새 데이터 유형이 추가되었습니다.
나누기(DIV) 페이지의 430	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
모듈로(MOD) 페이지의 437	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
곱하기(MUL) 페이지의 444	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
부정(NEG) 페이지의 452	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.

항목 이름	이유
제곱근(SQR/SQRT) 페이지의 458	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
빼기(SUB) 페이지의 465	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
부울 논리곱(BAND) 페이지의 504	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
부울 배타적 논리합(BXOR) 페이지의 510	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
부울 논리부정(BNOT) 페이지의 515	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
부울 논리합(BOR) 페이지의 519	새 데이터 유형과 새 펑션 블록 다이어그램 기능 언어가 추가되었습니다.
파일 검색 및 비교(FSC) 페이지의 581	설명 섹션의 .POS 비트가 .POS 로 변경되었습니다. 유효한 연산자 표가 제거되고 유효한 연산자 항목의 링크로 대체되었습니다.
파일 산술 및 로직(FAL) 페이지의 556	유효한 연산자 표가 제거되고 유효한 연산자 항목의 링크로 대체되었습니다.
유효한 연산자 페이지의 407	적용 가능한 명령어의 열과 행에 허용됨이 포함되도록 표가 업데이트되었습니다.
반복(FOR) 페이지의 730	루프 종료에 대한 설명이 업데이트되었습니다.
비례-적분-미분(PID) 페이지의 767	동작을 제어하는 .CA 비트의 .CTL 니모닉 설명이 업데이트되었습니다(0 = 역방향(SP-PV), 1 = 직접(PV- SP)).
사용권 유효성 검사(LV) 페이지의 959	새 명령어.
공통 특성 페이지의 963	기본 데이터 유형 항목에 대한 링크가 추가되었습니다.

항목 이름	이유
즉시 값 페이지의 967	정수 즉시 값 및 부동 소수점 즉시 값 표가 추가되었습니다.
데이터 변환 페이지의 967	최적의 데이터 유형이 즉시 데이터 유형으로 변경되고 확장 데이터 유형 USINT, INT, UINT, UDINT, ULINT, LREAL 이 포함되었습니다. SINT 또는 INT 를 DINT 로 변환 섹션에 DINT 를 LINT 로 변환이 추가되었습니다. 32 비트 및 64 비트 데이터 변환이 포함되었습니다.
기본 데이터 유형 페이지의 972	항목 제목이 데이터 유형에서 기본 데이터 유형으로 변경되었습니다. LINT, USINT, UINT, UDINT, ULINT, REAL 및 LREAL 이 추가되었습니다.
LINT 데이터 유형 페이지의 975	명령어에 사용되는 LINT 데이터 유형을 지원하는 적용 가능한 컨트롤러 목록이 추가되었습니다.
부동 소수점 값 페이지의 976	적용 가능한 컨트롤러 목록이 추가되었습니다. LREAL 태그 설명이 추가되었습니다.
배열을 통한 인덱스 페이지의 978	Logix Designer 에서 데이터 유형 태그만 확장된 첨자를 허용한다는 내용을 설명하는 2 개의 새로운 팁이 추가되었습니다. 사용 가능한 모든 정수 기본 데이터 유형을 첨자 인덱스로 사용할 수 있다는 설명도 추가되었습니다.
비트 주소 지정 페이지의 980	새 정의가 추가되었습니다.
FOR DO 페이지의 1016	루프 종료에 대한 설명이 업데이트되었습니다.

본 표시기로 명령어별 Logix5000 컨트롤러 사용 설명서를 찾을 수 있습니다.

Logix5000 Controller 일반 명령 참조 매뉴얼 1756-RM003	**Logix5000 컨트롤러 고급 프로세스 제어 및 드라이브/장비 위상 시퀀스 명령어 참조 매뉴얼 1756-RM006**	**Logix5000 Controllers Motion Instructions Reference Manual MOTION-RM002**
절대값(ABS)	알람(ALM)	마스터 구동 조정 제어(MDCC)
더하기(ADD)	장비 단계에 첨부(PATT)	모션 적용 축 튜닝(MAAT)
아날로그 알람(ALMA)	장비 시퀀스에 첨부(SATT)	모션 적용 후크업 진단(MAHD)
항상 거짓(AFI)	조정 제어(CC)	모션 아밍 출력 캠(MAOC)
아크 코사인(ACS, ACOS)	D 플립플롭(DFF)	모션 아밍 등록(MAR)
아크 사인(ASN, ASIN)	데드타임(DEDT)	모션 아밍 감시(MAW)
아크 탄젠트(ATN, ATAN)	미분(DERV)	모션 축 폴트 리셋(MAFR)
버퍼의 ASCII 문자(ACB)	장비 단계에서 분리(PDET)	모션 축 기어(MAG)
ASCII 버퍼 지우기(ACL)	장비 시퀀스에서 분리(SDET)	모션 축 홈(MAH)
ASCII 핸드셰이크 라인(AHL)	이산 3-상태 장치(D3SD)	모션 축 조그(MAJ)
ASCII 읽기(ARD)	이산 2-상태 장치(D2SD)	모션 축 이동(MAM)
ASCII 읽기 라인(ARL)	향상된 PID(PIDE)	모션 축 위치 캠(MAPC)
버퍼 라인에 대한 ASCII 테스트(ABL)	고급 선택(ESEL)	모션 축 정지(MAS)
ASCII 쓰기(AWT)	장비 단계 지우기 실패(PCLF)	모션 축 시간 캠(MATC)
ASCII 쓰기 추가(AWA)	장비 단계 명령(PCMD)	모션 축 종료(MASD)
비트 필드 분산(BTD)	장비 단계 외부 요청(PXRQ)	모션 축 종료 리셋(MASR)
비트 필드 분산(대상 포함)(BTDT)	장비 단계 실패(PFL)	모션 계산 캠 프로파일(MCCP)
비트 왼쪽 자리 이동(BSL)	장비 단계 새 파라미터(PRNP)	모션 조정 경로 이동(MCPM)
비트 오른쪽 자리 이동(BSR)	장비 단계 오버라이드 명령(POVR)	모션 계산 슬레이브 값(MCSV)
비트별 논리곱(AND)	장비 단계 일시 중지(PPD)	방향이 있는 모션 조정 변환(MCTO)
비트별 논리부정(NOT)	장비 시퀀스 할당 시퀀스 식별자(SASI)	모션 계산 변환 위치(MCTP)

Logix5000 Controller 일반 명령 참조 매뉴얼 1756-RM003	Logix5000 컨트롤러 고급 프로세스 제어 및 드라이브/장비 위상 시퀀스 명령어 참조 매뉴얼 1756-RM006	Logix5000 Controllers Motion Instructions Reference Manual MOTION-RM002
비트별 논리합(OR)	장비 시퀀스 지우기 실패(SCLF)	방향이 있는 모션 계산 변환 위치(MCTPO)
부울 논리곱(BAND)	장비 시퀀스 명령(SCMD)	모션 변경 다이내믹(MCD)
부울 배타적 논리합(BXOR)	장비 시퀀스 오버라이드(SOVR)	모션 조정 변경 다이내믹(MCCD)
부울 논리부정(BNOT)	함수 생성기(FGEN)	모션 조정 원형 이동(MCCM)
부울 논리합(BOR)	고역 통과 필터(HPF)	모션 조정 직선 이동(MCLM)
중단(BRK)	상한/하한(HLL)	모션 조정 종료(MCSD)
중단점(BPT)	적분기(INTG)	모션 조정 종료 리셋(MCSR)
지우기(CLR)	내부 모델 제어(IMC)	모션 조정 정지(MCS)
비교(CMP)	JK 플립플롭(JKFF)	모션 조정 변환(MCT)
BCD 로 변환(TOD)	리드-래그(LDLG)	모션 직접 드라이브 끄기(MDF)
정수로 변환(FRD)	저역 통과 필터(LPF)	모션 직접 드라이브 켜기(MDO)
파일 복사(COP), 동기식 복사 파일(CPS)	최대 캡처(MAXC)	모션 직접 시작(MDS)
코사인(COS)	최소 캡처(MINC)	모션 아밍 중지 출력 캠(MDOC)
계산(CPT)	모듈식 다변수 제어(MMC)	모션 아밍 중지 등록(MDR)
아래로 카운트(CTD)	이동 평균(MAVE)	모션 아밍 중지 감시(MDW)
위로 카운트(CTU)	이동 표준 편차(MSTD)	모션 그룹 종료(MGSD)
위로/아래로 카운트 CTUD	멀티플렉서(MUX)	모션 그룹 종료 리셋(MGSR)
데이터 전환(DTR)	노치 필터(NTCH)	모션 그룹 정지(MGS)
도(DEG)	단계 상태 완료(PSC)	모션 그룹 스트로브 위치(MGSP)
진단 감지(DDT)	위치 비례(POSP)	모션 재정의 위치(MRP)
디지털 알람(ALMD)	비례 + 적분(PI)	모션 실행 축 튜닝(MRAT)
DINT -> 문자열(DTOS)	펄스 승수(PMUL)	모션 실행 후크업 진단(MRHD)
나누기(DIV)	램프/소크(RMPS)	모션 서보 해제(MSF)
전환 끝(EOT)	속도 제한기(RLIM)	모션 서보 설정(MSO)
같음(EQU)	리셋 우세(RESD)	
파일 연산(FAL)	스케일(SCL)	

Logix5000 Controller 일반 명령 참조 매뉴얼 1756-RM003	Logix5000 컨트롤러 고급 프로세스 제어 및 드라이브/장비 위상 시퀀스 명령어 참조 매뉴얼 1756-RM006	Logix5000 Controllers Motion Instructions Reference Manual MOTION-RM002
파일 비트 비교(FBC)	S-곡선(SCRV)	
FIFO 로드(FFL)	2 차 컨트롤러(SOC)	
FIFO 언로드(FFU)	2 차 리드 래그(LDL2)	
파일 평균(AVE)	선택(SEL)	
파일 표준 편차(STD)	선택 부정(SNEG)	
파일 채우기(FLL)	선택한 합산기(SSUM)	
파일 정렬(SRT)	설정 우세(SETD)	
문자열 찾기(FIND)	분할 범위 시간 비례(SRTP)	
반복(FOR)	총계 계산기(TOT)	
파일 검색 및 비교(FSC)	위/아래 누적기(UPDN)	
시스템 값 가져오기(GSV) 및 시스템 값 설정(SST)		
크거나 같음(GEQ)		
큼(GRT)		
문자열 삽입(INSERT)		
즉시 출력(IOT)		
라벨로 이동(JMP) 및 라벨(LBL)		
서브루틴으로 이동(JSR), 서브루틴(SBR), 및 반환(RET)		
외부 루틴으로 이동(JXR)		
작음(LES)		
작거나 같음(LEQ)		
LIFO 로드(LFL)		
LIFO 언로드(LFU)		
validator 라이선스(LV)		
제한(LIM)		
로그 밑수(LOG)		
소문자(LOWER)		

Logix5000 Controller 일반 명령 참조 매뉴얼 1756-RM003	Logix5000 컨트롤러 고급 프로세스 제어 및 드라이브/장비 위상 시퀀스 명령어 참조 매뉴얼 1756-RM006	Logix5000 Controllers Motion Instructions Reference Manual MOTION-RM002
마스킹된 이동(MVM)		
마스킹된 이동(대상 포함)(MVMT)		
마스터 제어 리셋(MCR)		
마스킹된 같음(MEQ)		
메세지(MSG)		
중간 문자열(MID)		
모듈로(MOD)		
이동(MOV)		
곱하기(MUL)		
자연 로그(LN)		
부정(NEG)		
같지 않음(NEQ)		
작업 없음(NOP)		
원샷(ONS)		
하강 에지 원샷(OSF)		
입력에서 하강 에지 원샷(OSFI)		
상승 에지 원샷(OSR)		
입력에서 상승 에지 원샷(OSRI)		
출력 전원 공급(OTE)		
출력 래치(OTL)		
출력 래치 해제(OTU)		
비례-적분-미분(PID)		
라디안(RAD)		
실수 -> 문자열(RTOS)		
리셋(RES)		
SFC 리셋(SFR)		
반환(RET)		
켜짐 적산 타이머(RTO)		

Logix5000 Controller 일반 명령 참조 매뉴얼 1756-RM003	Logix5000 컨트롤러 고급 프로세스 제어 및 드라이브/장비 위상 시퀀스 명령어 참조 매뉴얼 1756-RM006	Logix5000 Controllers Motion Instructions Reference Manual MOTION-RM002
리셋 포함 켜짐 적산 타이머(RTOR)		
SFC 일시 중지(SFP)		
요소의 크기(SIZE)		
시퀀서 입력(SQI)		
시퀀서 로드(SQL)		
시퀀서 출력(SQO)		
사인(SIN)		
제곱근(SQR/SQRT)		
문자열 연결(CONCAT)		
문자열 삭제(DELETE)		
문자열 -> DINT(STOD)		
문자열 -> REAL(STOR)		
바이트 스왑(SWPB)		
빼기(SUB)		
탄젠트(TAN)		
꺼짐 지연 타이머(TOF)		
리셋 포함 꺼짐 지연 타이머(TOFR)		
켜짐 지연 타이머(TON)		
리셋 포함 켜짐 지연 타이머(TONR)		
일시 종료(TND)		
추적점(TPT)		
트리거 이벤트 태스크(EVENT)		
자르기(TRN)		
알 수 없는 명령어(UNK)		
대문자(UPPER)		

Logix5000 Controller 일반 명령 참조 매뉴얼 1756-RM003	Logix5000 컨트롤러 고급 프로세스 제어 및 드라이브/장비 위상 시퀀스 명령어 참조 매뉴얼 1756-RM006	Logix5000 Controllers Motion Instructions Reference Manual MOTION-RM002
사용자 중단 비활성화(UID)/사용자 중단 활성화(UIE)		
X 의 Y 제곱(XPY)		
닫힘 검사(XIC)		
열림 검사(XIO)		
비트별 배타적 논리합(XOR)		

4 장

입력/출력

5 장

비교 명령어

6 장

계산/연산 명령어

7 장

이동/로직 명령어

8 장

배열(파일)/기타 명령어

9 장

배열(파일)/이동 명령어

10 장

시퀀서 명령어

11 장

프로그램 제어 명령어

12 장

반복/중단 명령어

13 장

특수 명령어

14 장

삼각법 명령어

15 장

고급 연산

16 장

연산 변환 명령어

17 장

ASCII 시리얼 포트 명령어

18 장

ASCII 문자열 명령어

19 장

ASCII 변환 명령어

20 장

디버그 명령어

21 장

사용권 명령어

22 장

일반 명령어의 공통 특성

23 장

펑션 블록 속성

24 장

ST(스트럭쳐드 텍스트) 프로그래밍

인덱스

본 메뉴얼에서는 Logix 기반 컨트롤러에 설정된 일반, 동작, 프로세스, 드라이브 명령어를 상세하게 설명합니다.

GuardLogix 컨트롤러를 이용하는 안전 애프리케이션을 설계하거나 프로그램하거나 문제해결을 할 경우GuardLogix Safety Application Instruction Set Safety Reference Manual, 발행 번호 1756-RM095을 참고하십시오.

이 매뉴얼은 LOGIX 5000 컨트롤러의 일반적인 프로그래밍 및 작동 절차를 보여주는 관련 매뉴얼 세트 중 하나입니다.

일반 절차 매뉴얼 전체 목록을 보려면 LOGIX 5000 Controllers Common Procedures Programming Manual (발행 번호 1756-PM001)을 참조하십시오.

LOGIX 5000 컨트롤러라는 용어는 LOGIX 5000 운영 체제를 기반으로 하는 모든 컨트롤러를 의미합니다.

Studio 5000 환경

Studio 5000 Automation Engineering & Design Environment®는 엔지니어링 및 설계 요소를 공통 환경에 결합합니다. 첫 번째 요소는 Studio 5000 Logix Designer® 응용 프로그램입니다. Logix Designer 응용 프로그램은 RSLogix 5000® 소프트웨어의 또 다른 브랜드명이며 디스크리트, 프로세스, 배치, 모션, 안전 및 드라이브 기반 솔루션을 위해 LOGIX 5000™ 컨트롤러를 프로그램하는 제품으로 계속 사용됩니다.

Studio 5000® 환경은 Rockwell 의 미래를 위한 토대입니다. Automation® 엔지니어링은 도구와 기능을 설계합니다. Studio 5000 환경은 설계 엔지니어가 제어 시스템의 모든 요소를 개발할 수 환경 중 하나입니다.

추가 리소스

다음 문서에는 Rockwell Automation 제품에 관한 추가 정보가 들어 있습니다.

리소스	설명
Industrial Automation Wiring and Grounding Guidelines, 발행 번호 1770-4.1	Rockwell Automation 산업 시스템 설치를 위한 일반 가이드라인을 제공합니다.
http://ab.rockwellautomation.com에서 사용할 수 있는 제품 인증 웹페이지	적합성 선언, 인증서 및 기타 인증 세부 사항을 제공합니다.

http://www.rockwellautomation.com/literature에서 문서를 보거나 다운로드하십시오. 기술 문서의 종이 사본을 주문하려면 현지 Rockwell Automation 대리점이나 영업 담당자에게 문의하십시오.

법적 고지 사항

저작권 공고

이 문서와 모든 관련 Rockwell Software 제품은 Rockwell Automation Technologies, Inc.에 저작권이 있습니다. Rockwell Automation Technologies, Inc.의 사전 서면 동의 없이 복제 및/또는 배포하는 행위는 엄격하게 금지됩니다. 자세한 내용은 사용권 계약을 참조하십시오.

EULA(최종 사용자 사용권 계약)

하드 드라이브에서 제품의 설치 폴더에 있는 License.rtf 파일을 열어 Rockwell Automation EULA(최종 사용자 사용권 계약)(이하 "EULA")를 볼 수 있습니다.

오픈 소스 라이센스

이 제품에 포함된 소프트웨어에는 하나 이상의 오픈 소스 라이센스에 따라 사용이 허가되는 저작권 소프트웨어가 들어 있습니다. 소프트웨어에 해당 라이센스의 사본이 포함되어 있습니다. 이 제품에 포함된 오픈 소스 패키지의 해당 소스 코드는 해당 웹 사이트에서 찾을 수 있습니다.

또는, Rockwell Automation 웹 사이트의 문의 양식을 통해 Rockwell Automation 에 연락하여 전체 해당 소스 코드를 얻을 수 있습니다. http://www.rockwellautomation.com/global/about-us/contact/contact.page 요청 텍스트의 일부로 "오픈 소스"를 포함하십시오.

이 제품에 사용된 모든 오픈 소스 소프트웨어의 전체 목록과 해당 라이센스는 릴리스 노트에 포함된 OPENSOURCE 폴더에서 찾을 수 있습니다. 이 라이센스의 기본 설치 위치는 C:\Program Files (x86)\Common Files\Rockwell\Help\<*Product*>\ReleaseNotes\OPENSOURCE\index.htm 입니다.

상표 공지

Allen-Bradley, ControlBus, ControlFLASH, Compact GuardLogix, Compact I/O, ControlLogix, CompactLogix, DCM, DH+, Data Highway Plus, DriveLogix, DPI, DriveTools, Explorer, FactoryTalk, FactoryTalk Administration Console, FactoryTalk Alarms and Events, FactoryTalk Batch, FactoryTalk Directory, FactoryTalk Security, FactoryTalk Services Platform, FactoryTalk View, FactoryTalk View SE, FLEX Ex, FlexLogix, FLEX I/O, Guard I/O, High Performance Drive, Integrated Architecture, Kinetix, Logix5000, LOGIX 5000, Logix5550, MicroLogix, DeviceNet, EtherNet/IP, PLC-2, PLC-3, PLC-5, PanelBuilder, PowerFlex, PhaseManager, POINT I/O, PowerFlex, Rockwell Automation, RSBizWare, Rockwell Software, RSEmulate, Historian, RSFieldbus, RSLinx, RSLogix, RSNetWorx for DeviceNet, RSNetWorx for EtherNet/IP, RSMACC, RSView, RSView32, Rockwell Software Studio 5000 Automation Engineering & Design Environment, Studio 5000 View Designer, SCANport, SLC, SoftLogix, SMC Flex, Studio 5000, Ultra 100, Ultra 200, VersaView, WINtelligent, XM, SequenceManager 는 Rockwell Automation, Inc.의 상표입니다.

여기에 언급되지 않은 모든 Rockwell Automation 로고, 소프트웨어 또는 하드웨어 제품도 Rockwell Automation, Inc 의 상표 또는 등록 상표입니다.

기타 상표

CmFAS Assistant, CmDongle, CmStick, CodeMeter, CodeMeter Control Center 및 WIBU 는 미국 및/또는 기타 국가에서 WIBU-SYSTEMS AG 의 상표입니다.

기타 모든 상표는 해당 소유주의 자산이며, 이를 확인합니다.

보증

본 제품은 제품 사용권에 따라 보증됩니다. 제품 성능은 시스템 구성, 실행 중인 응용 프로그램, 작업자 제어, 유지관리 및 기타 관련 요인의 영향을 받을 수 있습니다. Rockwell Automation 은(는) 이러한 간섭 요인에 대해 책임을 지지 않습니다. 본 문서의 지침은 설명한 장비, 절차 또는 프로세스의 모든 형태나 모든 세부 사항을 다루지 않으며 설치, 작동 또는 유지관리 중에 발생 가능한 모든 우발적 상황에 맞는 지침도 제공하지 않습니다. 본 제품의 구현은 사용자마다 다를 수 있습니다.

본 문서는 제품 출시 당시 최신이지만 관련 소프트웨어는 출시 후 변경되었을 수 있습니다. Rockwell Automation, Inc.는 사전 고지 없이 언제든지 본 문서나 소프트웨어에 포함된 정보를 변경할 권한을 가지고 있습니다. 본 제품을 설치하거나 사용할 때 Rockwell 로부터 최신 정보를 입수할 책임은 귀하에게 있습니다.

환경 규정 준수

Rockwell Automation 은(는) 당사 웹 사이트(http://www.rockwellautomation.com/rockwellautomation/about-us/sustainability-ethics/product-environmental-compliance.page)에서 제품에 관한 최신 환경 정보를 제공합니다.

Rockwell 연락처

고객 지원 전화 - 1.440.646.3434

온라인 지원 - http://www.rockwellautomation.com/support/

알람 명령어

알람 명령어

알람 명령어를 사용하여 알람 조건을 모니터링 및 제어합니다.

Logix 기반 알람 명령어는 RSView® SE 응용 프로그램과 LOGIX 5000™ 컨트롤러 간에 알람 기능을 통합합니다.

사용 가능한 명령어

래더 다이어그램

ALMD	ALMA	ASO

펑션 블록

ALMD	ALMA

ST(스트럭쳐드 텍스트)

ALMD	ALMA	ASO

다음의 경우:	사용할 명령어:
래더 다이어그램, 펑션 블록 또는 ST(스트럭쳐드 텍스트)용 이산 부울 값 알람 기능을 제공할 경우	디지털 알람(ALMD) 명령어.
래더 다이어그램, 펑션 블록 다이어그램 또는 ST(스트럭쳐드 텍스트)용 아날로그 신호 레벨 및 변화율 알람 기능을 제공할 경우	아날로그 알람(ALMA) 명령어.
지정된 알람 집합의 모든 알람 조건에 대해 지정된 동작을 지시하려는 경우	알람 집합 동작(ASO) 명령어.

추가 참조

배열(파일)/기타 명령어 페이지의 545

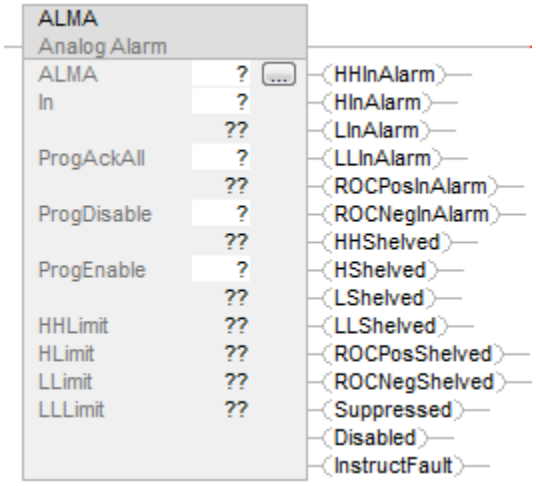
ASCII 변환 명령어 페이지의 927

아날로그 알람(ALMA)

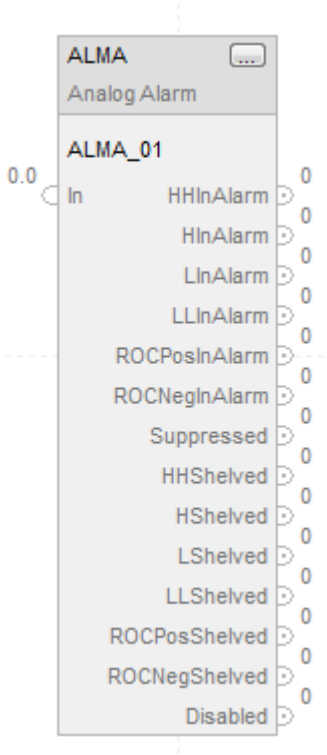
이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

ALMA 명령어는 아날로그 신호의 레벨 및 변화율에 대한 알람 기능을 제공합니다.

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

ALMA (ALMA,In,ProgAckAll,ProgDisable,ProgEnable)

피연산자

래더 다이어그램

피연산자	유형	형식	설명
ALMA	ALARM_ANALOG	구조	ALMA 구조
In	REAL DINT INT SINT	태그 즉시	알람 조건을 감지하는 알람 제한과 비교할 알람 입력 값.
ProgAckAll	BOOL	태그 즉시	거짓에서 참으로 전환 후 확인이 필요한 모든 알람 조건이 확인됩니다.
ProgDisable	BOOL	태그 즉시	참일 경우 알람이 비활성화됩니다(활성화 명령을 오버라이드하지 않음).

ProgEnable	BOOL	태그 즉시	참일 경우 알람이 활성화됩니다(비활성화 명령보다 우선 적용됨).

평션 블록

피연산자	유형	형식	설명
ALMA tag	ALARM_ANALOG	구조	ALMA 구조

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
ALMA	ALARM_ANALOG	구조	ALMA 구조
In	REAL DINT INT SINT	태그 즉시	알람 조건을 감지하는 알람 제한과 비교할 알람 입력 값.
ProgAckAll	BOOL	태그 즉시	거짓에서 참으로 전환 후 확인이 필요한 모든 알람 조건이 확인됩니다.
ProgDisable	BOOL	태그 즉시	참일 경우 알람이 비활성화됩니다(활성화 명령을 오버라이드하지 않음).
ProgEnable	BOOL	태그 즉시	참일 경우 알람이 활성화됩니다(비활성화 명령보다 우선 적용됨).

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 ST(스트럭처드 텍스트) 구문을 참조하십시오.

ALMA 구조

입력 파라미터

입력 파라미터	데이터 유형	설명
활성화 입력	BOOL	래더 다이어그램: 렁 상태에 해당합니다. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. ST(스트럭쳐드 텍스트): 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다. 펑션 블록: 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
In	REAL	알람 조건을 감지하는 알람 제한과 비교할 알람 입력 값. 기본값 = 0.0. 래더 다이어그램: 명령어 피연산자에서 복사됩니다. ST(스트럭쳐드 텍스트): 명령어 피연산자에서 복사됩니다.
InFault	BOOL	입력에 불량 상태 표시기. 사용자 응용 프로그램에서 InFault 가 입력 신호에 에러가 있음을 나타내도록 설정할 수 있습니다. 설정된 경우 명령어는 InFaulted (Status.1)를 설정하고, 거짓으로 해제된 경우 명령어에 의해 InFaulted (Status.1)가 거짓으로 해제됩니다. 어느 경우이든 명령어는 In 에서 알람 조건을 계속 평가합니다. 기본값은 거짓(올바른 상태)입니다.
HHEnabled	BOOL	상-상한 알람 조건 감지. 상-상한 알람 조건 감지를 활성화하려면 참으로 설정합니다. 상-상한 알람 조건 감지를 사용하지 않으려면 거짓으로 해제합니다. 기본값은 설정 상태입니다.

입력 파라미터	데이터 유형	설명
HEnabled	BOOL	상한 알람 조건 감지. 상한 알람 조건 감지를 활성화하려면 참으로 설정합니다. 상한 알람 조건 감지를 사용하지 않으려면 거짓으로 해제합니다. 기본값은 설정 상태입니다.
LEnabled	BOOL	하한 알람 조건 감지. 하한 알람 조건 감지를 활성화하려면 참으로 설정합니다. 하한 알람 조건 감지를 사용하지 않으려면 거짓으로 해제합니다. 기본값은 설정 상태입니다.
LLEnabled	BOOL	하-하한 알람 조건 감지. 하-하한 알람 조건 감지를 활성화하려면 참으로 설정합니다. 하-하한 알람 조건 감지를 사용하지 않으려면 거짓으로 해제합니다. 기본값은 설정 상태입니다.
AckRequired	BOOL	알람 확인이 필요한지 여부를 지정합니다. 참으로 설정된 경우 확인이 필요합니다. 거짓으로 해제된 경우 확인이 필요하지 않으며, HHAcked, HAcked, LAcked, LLAcked, ROCPosAcked, ROCNegAcked 가 항상 참으로 설정됩니다. 기본값은 참입니다.
ProgAckAll	BOOL	모든 알람 조건을 확인하도록 사용자 프로그램에 의해 참으로 설정됩니다. 알람 조건이 확인되지 않은 경우에만 적용됩니다. 거짓 - 참 전환이 필요합니다. 기본값은 거짓입니다. 래더 다이어그램: 명령어 피연산자에서 복사됩니다. ST(스트럭쳐드 텍스트): 명령어 피연산자에서 복사됩니다.
OperAckAll	BOOL	모든 알람 조건을 확인하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 조건이 확인되지 않은 경우에만 적용됩니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
HHProgAck	BOOL	상-상한 프로그램 확인. 상-상한 조건을 확인하도록 사용자 프로그램에 의해 참으로 설정됩니다. 알람 조건이 확인되지 않은 경우에만 적용됩니다. 거짓 - 참 전환이 필요합니다. 기본값은 거짓입니다.

입력 파라미터	데이터 유형	설명
HHOperAck	BOOL	상-상한 작업자 확인. 상-상한 조건을 확인하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 조건이 확인되지 않은 경우에만 적용됩니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
HProgAck	BOOL	상한 프로그램 확인. 상한 조건을 확인하도록 사용자 프로그램에 의해 참으로 설정됩니다. 알람 조건이 확인되지 않은 경우에만 적용됩니다. 거짓 - 참 전환이 필요합니다. 기본값은 거짓입니다.
HOperAck	BOOL	상한 작업자 확인. 상한 조건을 확인하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 조건이 확인되지 않은 경우에만 적용됩니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
LProgAck	BOOL	하한 프로그램 확인. 하한 조건을 확인하도록 사용자 프로그램에 의해 참으로 설정됩니다. 알람 조건이 확인되지 않은 경우에만 적용됩니다. 거짓 - 참 전환이 필요합니다. 기본값은 거짓입니다.
LOperAck	BOOL	하한 작업자 확인. 하한 조건을 확인하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 조건이 확인되지 않은 경우에만 적용됩니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
LLProgAck	BOOL	하-하한 프로그램 확인. 하-하한 조건을 확인하도록 사용자 프로그램에 의해 참으로 설정됩니다. 알람 조건이 확인되지 않은 경우에만 적용됩니다. 거짓 - 참 전환이 필요합니다. 기본값은 거짓입니다.
LLOperAck	BOOL	하-하한 작업자 확인. 하-하한 조건을 확인하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 조건이 확인되지 않은 경우에만 적용됩니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.

입력 파라미터	데이터 유형	설명
ROCPosProgAck	BOOL	양의 변화율 프로그램 확인. 양의 변화율 조건을 확인하도록 사용자 프로그램에 의해 참으로 설정됩니다. 알람 조건이 확인되지 않은 상태에서 거짓 - 참 전환이 필요합니다. 기본값은 거짓입니다.
ROCPosOperAck	BOOL	양의 변화율 작업자 확인. 양의 변화율 조건을 확인하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 조건이 확인되지 않은 상태에서 거짓 - 참 전환이 필요합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 설정됩니다. 기본값은 거짓입니다.
ROCNegProgAck	BOOL	음의 변화율 프로그램 확인. 음의 변화율 조건을 확인하도록 사용자 프로그램에 의해 참으로 설정됩니다. 알람 조건이 확인되지 않은 상태에서 거짓 - 참 전환이 필요합니다. 기본값은 거짓입니다.
ROCNegOperAck	BOOL	음의 변화율 작업자 확인. 음의 변화율 조건을 확인하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 조건이 확인되지 않은 상태에서 거짓 - 참 전환이 필요합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
ProgSuppress	BOOL	알람을 억제하도록 사용자 프로그램에 의해 참으로 설정됩니다. 기본값은 해제 상태입니다.
OperSuppress	BOOL	알람을 억제하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
ProgUnsuppress	BOOL	알람을 억제 해제하도록 사용자 프로그램에 의해 참으로 설정됩니다. 억제 명령보다 우선 순위를 갖습니다. 기본값은 거짓입니다.

입력 파라미터	데이터 유형	설명
OperUnsuppress	BOOL	알람을 억제 해제하기 위해 작업자 인터페이스에서 참으로 설정합니다. 억제 명령보다 우선 순위를 갖습니다. 알람 명령어에 의해 이 파라미터가 거짓으로 설정됩니다. 기본값은 거짓입니다.
HHOperShelve	BOOL	상-상한 작업자 보류. 상-상한 조건을 보류하거나 재보류하기 위해 작업자 인터페이스에서 참으로 설정합니다. 거짓 - 참 전환이 필요합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다. 보류 해제 명령이 보류 명령보다 우선합니다. 알람을 보류하면 알람 처리가 연기됩니다. 보류는 시간이 제한된다는 점만 제외하고 알람 억제와 유사합니다. 알람이 보류된 동안 확인될 경우 알람이 다시 활성화되더라도 계속 확인되어 있습니다. 보류 기간이 종료되면 알람이 확인되지 않은 상태가 됩니다.
HOperShelve	BOOL	상한 작업자 보류. 상한 조건을 보류하거나 재보류하기 위해 작업자 인터페이스에서 참으로 설정합니다. 한 프로그램 스캔은 거짓, 다음 프로그램 스캔은 참으로 전환되어야 합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다. 보류 해제 명령이 보류 명령보다 우선합니다.
LOperShelve	BOOL	하한 작업자 보류. 하한 조건을 보류하거나 재보류하기 위해 작업자 인터페이스에서 참으로 설정합니다. 한 프로그램 스캔은 거짓, 다음 프로그램 스캔은 참으로 전환되어야 합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다. 보류 해제 명령이 보류 명령보다 우선합니다.

입력 파라미터	데이터 유형	설명
LLOperShelve	BOOL	하-하한 작업자 보류. 하-하한 조건을 보류하거나 재보류하기 위해 작업자 인터페이스에서 참으로 설정합니다. 한 프로그램 스캔은 거짓, 다음 프로그램 스캔은 참으로 전환되어야 합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다. 보류 해제 명령이 보류 명령보다 우선합니다.
ROCPosOperShelve	BOOL	양의 변화율 작업자 보류. 양의 변화율 조건을 보류하거나 재보류하기 위해 작업자 인터페이스에서 참으로 설정합니다. 한 프로그램 스캔은 거짓, 다음 프로그램 스캔은 참으로 전환되어야 합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다. 보류 해제 명령이 보류 명령보다 우선합니다.
ROCNegOperShelve	BOOL	음의 변화율 작업자 보류. 음의 변화율 조건을 보류하거나 재보류하기 위해 작업자 인터페이스에서 참으로 설정합니다. 한 프로그램 스캔은 거짓, 다음 프로그램 스캔은 참으로 전환되어야 합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다. 보류 해제 명령이 보류 명령보다 우선합니다.
ProgUnshelveAll	BOOL	이 알람의 모든 조건을 보류 해제하도록 사용자 프로그램에 의해 참으로 설정됩니다. 보류 및 보류 해제 모두 참인 경우 보류 해제 명령이 보류 명령보다 우선 적용됩니다. 기본값은 거짓입니다.
HHOperUnshelve	BOOL	상-상한 작업자 보류 해제. 상-상한 조건을 보류 해제하려면 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 보류 및 보류 해제 모두 참인 경우 보류 해제 명령이 보류 명령보다 우선 적용됩니다. 기본값은 거짓입니다.

입력 파라미터	데이터 유형	설명
HOperUnshelve	BOOL	상한 작업자 보류 해제. 상한 조건을 보류 해제하려면 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 보류 및 보류 해제 모두 참인 경우 보류 해제 명령이 보류 명령보다 우선 적용됩니다. 기본값은 거짓입니다.
LOperUnshelve	BOOL	하한 작업자 보류 해제. 하한조건을 보류 해제하려면 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 보류 및 보류 해제 모두 참인 경우 보류 해제 명령이 보류 명령보다 우선 적용됩니다. 기본값은 거짓입니다.
LLOperUnshelve	BOOL	하-하한 작업자 보류 해제. 하-하한 조건을 보류 해제하려면 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 보류 및 보류 해제 모두 참인 경우 보류 해제 명령이 보류 명령보다 우선 적용됩니다. 기본값은 거짓입니다.
ROCPosOperUnshelve	BOOL	양의 변화율 작업자 보류 해제. 양의 변화율 조건을 보류 해제하려면 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 보류 및 보류 해제가 둘 다 설정된 경우 보류 해제 명령이 보류 명령보다 우선 적용됩니다. 기본값은 거짓입니다.
ROCNegOperUnshelve	BOOL	음의 변화율 작업자 보류 해제. 음의 변화율 조건을 보류 해제하려면 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 보류 및 보류 해제 모두 참인 경우 보류 해제 명령이 보류 명령보다 우선 적용됩니다. 기본값은 거짓입니다.
ProgDisable	BOOL	명령어 피연산자에서 복사됩니다.

입력 파라미터	데이터 유형	설명
OperDisable	BOOL	알람을 비활성화하려면 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
ProgEnable	BOOL	명령어 피연산자에서 복사됩니다.
OperEnable	BOOL	알람을 활성화하려면 작업자 인터페이스에서 이 파라미터를 참으로 설정합니다. 사용 안 함 명령보다 우선 순위를 갖습니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
AlarmCountReset	BOOL	모든 조건의 알람 카운트가 리셋되도록 작업자 인터페이스에 의해 참으로 설정됩니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
HHMinDurationEnable	BOOL	상-상한 최소 기간 활성화. 상-상한 조건이 감지될 때 최소 기간 타이머를 활성화하려면 이 파라미터를 참으로 설정합니다. 기본값은 참입니다.
HMinDurationEnable	BOOL	상한 최소 기간 활성화. 상한 조건이 감지될 때 최소 기간 타이머를 활성화하려면 이 파라미터를 참으로 설정합니다. 기본값은 참입니다.
LMinDurationEnable	BOOL	하한 최소 기간 활성화. 하한 조건이 감지될 때 최소 기간 타이머를 활성화하려면 이 파라미터를 참으로 설정합니다. 기본값은 참입니다.
LLMinDurationEnable	BOOL	하-하한 최소 기간 활성화. 하-하한 조건이 감지될 때 최소 기간 타이머를 활성화하려면 이 파라미터를 참으로 설정합니다. 기본값은 참입니다.
HHLimit	REAL	상-상한 알람 제한 유효값 = HLimit < HHLimit < 최대 양의 부동 소수점 수 기본값 = 0.0.

입력 파라미터	데이터 유형	설명
HHSeverity	DINT	상-상한 알람 조건의 심각도. 이는 컨트롤러의 알람 처리에 영향을 주지는 않지만 알람 구독자 측에서 함수를 정렬 및 필터링하는 데 사용할 수 있습니다. 유효값 = 1 ~ 1000(1000 = 가장 심각함; 1 = 가장 심각하지 않음) 기본값 = 500.
HLimit	REAL	상한 알람 제한 유효값 = LLimit < HLimit < HHLimit. 기본값 = 0.0.
HSeverity	DINT	상한 알람 조건의 심각도. 이는 컨트롤러의 알람 처리에 영향을 주지는 않지만 알람 구독자 측에서 함수를 정렬 및 필터링하는 데 사용할 수 있습니다. 유효값 = 1 ~ 1000(1000 = 가장 심각함; 1 = 가장 심각하지 않음) 기본값 = 500.
LLimit	REAL	하한 알람 제한 유효값 = LLLimit < LLimit < HLimit. 기본값 = 0.0.
LSeverity	DINT	하한 알람 조건의 심각도. 이는 컨트롤러의 알람 처리에 영향을 주지는 않지만 알람 구독자 측에서 함수를 정렬 및 필터링하는 데 사용할 수 있습니다. 유효값 = 1 ~ 1000(1000 = 가장 심각함; 1 = 가장 심각하지 않음) 기본값 = 500.
LLLimit	REAL	하-하한 알람 제한. 유효값 = 최대 음의 부동 소수점 수 < LLLimit < LLimit. 기본값 = 0.0.
LLSeverity	DINT	하-하한 알람 조건의 심각도. 이는 컨트롤러의 알람 처리에 영향을 주지는 않지만 알람 구독자 측에서 함수를 정렬 및 필터링하는 데 사용할 수 있습니다. 유효값 = 1 ~ 1000(1000 = 가장 심각함; 1 = 가장 심각하지 않음) 기본값 = 500.

입력 파라미터	데이터 유형	설명
MinDurationPRE	DINT	조건이 InAlarm 으로 표시되고 클라이언트에 알람 알림이 전송되기 전에 알람 레벨 조건이 참으로 유지되어야 하는 미리 설정된 최소 지속 시간(밀리초). 컨트롤러는 알람 조건이 감지되는 즉시 알람 데이터를 수집하므로 최소 기간에 도달할 때까지 기다리는 동안 데이터가 손실되지 않습니다. 변화율 조건이나 최소 기간 감지가 사용되지 않는 조건에는 적용되지 않습니다. MinDurationPRE 는 정상 레벨에서 양 또는 음의 방향으로 첫 번째 운동 거리에만 적용됩니다. 예를 들어 상한 조건이 시간 초과되면 상-상한 조건이 즉시 활성화되고 해당 타임아웃 기간 동안 하한 조건이 대기합니다. 유효값 = 0 ~ 2147483647 기본값 = 0.
ShelveDuration	DINT	보류된 알람이 보류되는 기간(분). 최소 시간은 1 분입니다. 최대 시간은 MaxShelveDuration 으로 정의됩니다.
MaxShelveDuration	DINT	알람을 보류할 수 있는 최대 기간(분).
Deadband	REAL	상-상한, 상한, 하한 및 하-하한 알람 레벨이 정상으로 돌아갔는지 확인하기 위한 불감대. 0 이 아닌 불감대는 In 값이 계속 변경되지만 레벨 조건 임계값에 가깝게 유지되는 경우 알람 조건 채터를 줄일 수 있습니다. Deadband 값은 InAlarm(활성) 상태로의 전환에 영향을 끼치지 않습니다. 레벨 조건이 활성화된 후 조건이 비활성(보통) 상태로 되돌아가기 전에 In 값은 다음 중 하나에 해당되어야 합니다. 임계값에서 불감대를 뺀 값보다 작아야 합니다(상한 및 상-상한 조건의 경우). 또는 임계값에 불감대를 더한 값보다 커야 합니다(하한 및 하-하한 조건의 경우). Deadband 는 최소 지속 시간 측정 조건을 평가하는 데 사용되지 않습니다. 유효값 = 0 ≤ Deadband < 첫 번째 활성화된 하한 알람에서 첫 번째 활성화된 상한 알람 사이의 범위 기본값 = 0.0.

입력 파라미터	데이터 유형	설명
ROCPosLimit	REAL	증가하는 변화율의 제한(초당 단위). 모든 값 > 0.0 이며 ROCPeriod 도 0.0 보다 클 경우 감지를 활성화합니다. 유효값 = 0.0 ~ 최대 가능 부동 소수점 수 기본값 = 0.0.
ROCPosSeverity	DINT	증가하는 변화율 조건의 심각도. 이는 컨트롤러의 알람 처리에 영향을 주지는 않지만 알람 구독자 측에서 함수를 정렬 및 필터링하는 데 사용할 수 있습니다. 유효값 = 1 ~ 1000(1000 = 가장 심각함; 1 = 가장 심각하지 않음) 기본값 = 500.
ROCNegLimit	REAL	감소하는 변화율의 제한(초당 단위). 모든 값 > 0.0 이며 ROCPeriod 도 0.0 보다 클 경우 감지를 활성화합니다. 유효값 = 0.0 ~ 최대 가능 부동 소수점 수 기본값 = 0.0.
ROCNegSeverity	DINT	감소하는 변화율 조건의 심각도. 이는 컨트롤러의 알람 처리에 영향을 주지는 않지만 알람 구독자 측에서 함수를 정렬 및 필터링하는 데 사용할 수 있습니다. 유효값 = 1 ~ 1000(1000 = 가장 심각함; 1 = 가장 심각하지 않음) 기본값 = 500.
ROCPeriod	REAL	변화율 값을 계산하기 위한 초 단위 기간(샘플링 주기). 샘플링 주기가 만료될 때마다 새로운 In 샘플이 저장되고 ROC 가 다시 계산됩니다. 변화율 감지는 아날로그 알람의 다른 조건과 같은 활성화 비트 대신 0 이 아닌 값을 ROCPeriod 에 놓으면 활성화됩니다. 유효값 = 0.0 ~ 32767.0 기본값 = 0.0.

출력 파라미터

다음은 래더 로직에 공통된 출력 파라미터입니다.

출력 파라미터	데이터 유형	설명
AnyInAlarmUnack	BOOL	알람이 활성화되고 확인된 상태. 알람 조건이 감지되고 확인되지 않은 경우 참으로 설정됩니다. 모든 알람 조건이 비활성이고 확인되었거나 둘 다에 해당하는 경우 거짓으로 해제됩니다.
HHInAlarm	BOOL	상-상한 알람 조건 상태. 상-상한 조건이 활성인 경우 참으로 설정되고, 상-상한조건이 없으면 거짓으로 해제됩니다.
HInAlarm	BOOL	상한 알람 조건 상태. 상한조건이 활성인 경우 참으로 설정됩니다. 상한 조건이 없으면 거짓으로 해제됩니다.
LInAlarm	BOOL	하한 알람 조건 상태. 하한 조건이 활성인 경우 참으로 설정되고, 하한 조건이 없으면 거짓으로 해제됩니다.
LLInAlarm	BOOL	하-하한 알람 조건. 하-하한 조건이 활성인 경우 참으로 설정되고, 하-하한 조건이 없으면 거짓으로 해제됩니다.
ROCPosInAlarm	BOOL	양의 변화율 알람 조건 상태. 양의 변화율조건이 있으면 참으로 설정되고, 양의 변화율조건이 없으면 거짓으로 해제됩니다.
ROCNegInAlarm	BOOL	음의 변화율 알람 조건 상태. 음의 변화율 조건이 있으면 참으로 설정되고, 음의 변화율 조건이 없으면 거짓으로 해제됩니다.
ROC	REAL	In 값의 계산된 변화율. 이 값은 각 ROCPeriod 가 경과한 후 명령어를 검색할 때마다 업데이트됩니다. ROC 값은 ROCPosInAlarm 및 ROCNegInAlarm 조건을 평가하는 데 사용됩니다. ROC = (현재 In 샘플 – 이전 In 샘플) / ROCPeriod
HHAcked	BOOL	상-상한 조건 확인된 상태. 상-상한 조건이 확인된 경우 참으로 설정됩니다. AckRequired 가 거짓으로 해제된 경우 항상 참으로 설정됩니다. 상-상한 조건이 확인되지 않은 경우 거짓으로 해제됩니다.
HAcked	BOOL	상한 알람 조건 확인된 상태. 상한 조건이 확인된 경우 참으로 설정됩니다. AckRequired 가 거짓으로 해제된 경우 항상 참으로 설정됩니다. 상한 조건이 확인되지 않은 경우 거짓으로 해제됩니다.
LAcked	BOOL	하한 조건 확인된 상태. 하한 조건이 확인된 경우 참으로 설정됩니다. AckRequired 가 거짓으로 해제된 경우 항상 참으로 설정됩니다. 하한 조건이 확인되지 않은 경우 거짓으로 해제됩니다.

출력 파라미터	데이터 유형	설명
LLAcked	BOOL	하-하한 조건 확인된 상태. 하-하한 조건이 확인된 경우 참으로 설정됩니다. AckRequired 가 거짓으로 해제된 경우 항상 참입니다. 하-하한 조건이 확인되지 않은 경우 거짓으로 해제됩니다.
ROCPosAcked	BOOL	양의 변화율 조건 확인된 상태. 양의 변화율 조건이 확인된 경우 참으로 설정됩니다. AckRequired 가 거짓으로 해제된 경우 항상 참입니다. 양의 변화율 조건이 확인되지 않은 경우 거짓으로 해제됩니다.
ROCNegAcked	BOOL	음의 변화율 조건 확인된 상태. 음의 변화율 조건이 확인된 경우 참으로 설정됩니다. AckRequired 가 거짓으로 해제된 경우 항상 참으로 설정됩니다. 음의 변화율 조건이 확인되지 않은 경우 거짓으로 해제됩니다.
HHInAlarmUnack	BOOL	상-상한 조건이 활성화되고 확인되지 않은 상태. 상-상한 조건이 활성(HHInAlarm 이 참)이고 확인되지 않은 경우 참으로 설정되고, 상-상한 조건이 비활성화이고 확인되었거나 둘 다에 해당하는 경우 거짓으로 해제됩니다.
HInAlarmUnack	BOOL	상한 조건이 활성화되고 확인되지 않은 상태. 상한 조건이 활성(HInAlarm 이 참)이고 확인되지 않은 경우 참으로 설정되고, 상한 조건이 비활성이고 확인되었거나 둘 다에 해당하는 경우 거짓으로 해제됩니다.
LInAlarmUnack	BOOL	하한 조건이 활성화되고 확인되지 않은 상태. 하한 조건이 활성(LInAlarm 이 참)이고 확인되지 않은 경우 참으로 설정되고, 하한 조건이 비활성이고 확인되었거나 둘 다에 해당하는 경우 거짓으로 해제됩니다.
LLInAlarmUnack	BOOL	하-하한 조건이 활성화되고 확인되지 않은 상태. 하-하한 조건이 활성(LLInAlarm 이 참)이고 확인되지 않은 경우 참으로 설정되고, 하-하한 조건이 비활성이고 확인되었거나 둘 다에 해당하는 경우 거짓으로 해제됩니다.

출력 파라미터	데이터 유형	설명
ROCPosInAlarmUnack	BOOL	양의 변화율 조건이 활성화되고 확인된 상태. 양의 변화율 조건이 활성(ROCPosInAlarm 이 참)이고 확인되지 않은 경우 참으로 설정되고, 양의 변화율 조건이 비활성이고 확인되었거나 둘 다에 해당하는 경우 거짓으로 해제됩니다.
ROCNegInAlarmUnack	BOOL	음의 변화율 조건이 활성화되고 확인된 상태. 음의 변화율 조건이 활성(ROCNegInAlarm 이 참)이고 확인되지 않은 경우 참으로 설정되고, 음의 변화율 조건이 비활성되고 확인되었거나 둘 다에 해당하는 경우 거짓으로 해제됩니다.
Suppressed	BOOL	알람 억제 상태. 알람이 억제된 경우 참으로 설정되고, 알람이 억제되지 않은 경우 거짓으로 해제됩니다.
HHShelved	BOOL	상-상한 조건 보류 상태. 상-상한 조건이 보류된 경우 참으로 설정되고, 상-상한 조건이 보류 해제되면 거짓으로 해제됩니다.
HShelved	BOOL	상한 조건 보류 상태. 상한 조건이 보류된 경우 참으로 설정되고, 상한 조건이 보류 해제되면 거짓으로 해제됩니다.
LShelved	BOOL	하한 조건 보류 상태. 하한 조건이 보류된 경우 참으로 설정되고, 하한 조건이 보류 해제되면 거짓으로 해제됩니다.
LLShelved	BOOL	하-하한 조건 보류 상태. 하-하한 조건이 보류된 경우 참으로 설정되고, 하-하한 조건이 보류 해제되면 거짓으로 해제됩니다.
ROCPosShelved	BOOL	양의 변화율 조건 보류 상태. 양의 변화율 조건이 보류된 경우 참으로 설정되고, 양의 변화율 조건이 보류 해제되면 거짓으로 해제됩니다.
ROCNegShelved	BOOL	음의 변화율 조건 보류 상태. 음의 변화율 조건이 보류된 경우 참으로 설정되고, 음의 변화율 조건이 보류 해제되면 거짓으로 해제됩니다.
비활성화	BOOL	알람 사용 안 함 상태. 알람을 사용할 수 없는 경우(비활성화됨) 참으로 설정되고, 알람이 활성화되면 거짓으로 해제됩니다.
Commissioned	BOOL	의뢰됨 비트는 사용되지 않습니다.
MinDurationACC	DINT	사용하지 않습니다. 값이 항상 0 입니다.

출력 파라미터	데이터 유형	설명
HHInAlarmTime	LINT	ALMA 명령어에서 In 값이 가장 최근 활성 상태로의 전환에 대한 상-상한 조건 제한을 초과했음을 감지할 때의 타임스탬프.
HHAlarmCount	DINT	상-상한 조건이 활성화된 횟수. 최고 값에 도달하면 카운터의 값이 최고 카운트 값으로 유지됩니다.
HInAlarmTime	LINT	ALMA 명령어에서 In 값이 가장 최근 활성 상태로의 전환에 대한 상한 조건 제한을 초과했음을 감지할 때의 타임스탬프.
HAlarmCount	DINT	상한 조건이 활성화된 횟수. 최고 값에 도달하면 카운터의 값이 최고 카운트 값으로 유지됩니다.
LInAlarmTime	LINT	ALMA 명령어에서 In 값이 가장 최근 활성 상태로의 전환에 대한 하한 조건 제한을 초과했음을 감지할 때의 타임스탬프.
LAlarmCount	DINT	하한 조건이 활성화된 횟수. 최고 값에 도달하면 카운터의 값이 최고 카운트 값으로 유지됩니다.
LLInAlarmTime	LINT	ALMA 명령어에서 In 값이 가장 최근 활성 상태로의 전환에 대한 하-하한 조건 제한을 초과했음을 감지할 때의 타임스탬프.
LLAlarmCount	DINT	하-하한 조건이 활성화된 횟수. 최고 값에 도달하면 카운터의 값이 최고 카운트 값으로 유지됩니다.
ROCPosInAlarmTime	LINT	ALMA 명령어에서 In 값이 가장 최근 활성 상태로의 전환에 대한 양의 변화율 조건 제한을 초과했음을 감지할 때의 타임스탬프.
ROCPosInAlarmCount	DINT	양의 변화율 조건이 활성화된 횟수. 최고 값에 도달하면 카운터의 값이 최고 카운트 값으로 유지됩니다.
ROCNegInAlarmTime	LINT	ALMA 명령어에서 In 값이 가장 최근 활성 상태로의 전환에 대한 음의 변화율 조건 제한을 초과했음을 감지할 때의 타임스탬프.
ROCNegAlarmCount	DINT	음의 변화율 조건이 활성화된 횟수. 최고 값에 도달하면 카운터의 값이 최고 카운트 값으로 유지됩니다.
AckTime	LINT	가장 최근 조건 확인의 타임스탬프. 알람에 확인이 필요하지 않은 경우 이 타임스탬프는 가장 최근의 조건 알람 시간과 같습니다.

<table>
<tr><th>출력 파라미터</th><th>데이터 유형</th><th>설명</th></tr>
<tr><td>RetToNormalTime</td><td>LINT</td><td>정상 상태로 돌아가는 알람의 타임스탬프</td></tr>
<tr><td>AlarmCountResetTime</td><td>LINT</td><td>알람 카운트가 초기화된 때를 나타내는 타임스탬프.</td></tr>
<tr><td>ShelveTime</td><td>LINT</td><td>알람 조건이 마지막으로 보류된 시간을 나타내는 타임스탬프. 알람 조건이 보류된 경우 컨트롤러에서 설정합니다. 알람 조건은 여러 번 보류 및 보류 해제할 수 있습니다. 알람 조건을 보류할 때마다 타임스탬프가 현재 시간으로 설정됩니다.</td></tr>
<tr><td>UnshelveTime</td><td>LINT</td><td>모든 알람 조건이 보류 해제될 시간을 나타내는 타임스탬프. 알람 조건이 아직 보류되지 않은 경우에만 값이 설정됩니다. 타임스탬프는 ShelveDuration 기간과 현재 시간의 합으로 결정됩니다. 알람 조건이 프로그래밍 방식 또는 작업자에 의해 보류 해제되지 않고 다른 알람 조건이 보류되지 않은 경우 값이 현재 시간으로 설정됩니다.</td></tr>
<tr><td>Status</td><td>DINT</td><td>혼합된 상태 표시:
<table>
<tr><th>상태 플래그</th><th>CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570 컨트롤러</th><th>CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러</th></tr>
<tr><td>Status.0 = InstructFault</td><td>X</td><td>X</td></tr>
<tr><td>Status.1= InFaulted</td><td>X</td><td>X</td></tr>
<tr><td>Status.2 = SeverityInv</td><td>X</td><td>X</td></tr>
<tr><td>Status.3 = AlarmLimitsInv</td><td>X</td><td>X</td></tr>
<tr><td>Status.4 = DeadbandInv</td><td>X</td><td>X</td></tr>
<tr><td>Status.5 = ROCPosLimitInv</td><td>X</td><td>X</td></tr>
<tr><td>Status.6 = ROCNegLimitInv</td><td>X</td><td>X</td></tr>
<tr><td>Status.7 = ROCPeriodInv</td><td>X</td><td>X</td></tr>
<tr><td>Status.8 = Overflow</td><td>-</td><td>X</td></tr>
</table></td></tr>
</table>

출력 파라미터	데이터 유형	설명
InstructFault (Status.0)	BOOL	명령어 에러 조건이 있습니다. 마이너 또는 메이저 컨트롤러 에러가 아닙니다. 나머지 상태 비트를 확인하여 발생한 에러를 판단합니다.
InFaulted (Status.1)	BOOL	사용자 프로그램에서 InFault 가 품질이 좋지 않은 입력 데이터를 나타내도록 설정했습니다. 알람은 알람 조건에 대해 In 평가를 계속합니다.
SeverityInv (Status.2)	BOOL	알람 심각도 구성이 잘못되었습니다. 심각도 < 1 이면 명령어는 Severity = 1 을 사용합니다. 심각도 >1000 이면 명령어는 Severity = 1000 을 사용합니다.
AlarmLimitsInv (Status.3)	BOOL	알람 제한 구성이 유효하지 않습니다(예: LLimit < LLLimit). 잘못된 경우 명령어는 모든 레벨 조건의 활성 비트를 해제하고, 폴트가 해결될 때까지 새로운 레벨 조건을 감지하지 않습니다.
DeadbandInv (Status.4)	BOOL	불감대 구성이 유효하지 않습니다. 유효하지 않을 경우 명령어는 Deadband = 0.0 을 사용합니다. 유효값 = 0 ≤ Deadband < 첫 번째 활성화된 하한 알람에서 첫 번째 활성화된 상한 알람 사이의 범위
ROCPosLimitInv (Status.5)	BOOL	양의 변화율 제한이 잘못되었습니다. 유효하지 않을 경우 명령어는 양의 변화율 감지를 사용하지 않도록 설정하는 ROCPosLimit = 0.0 을 사용합니다.
ROCNegLimitInv (Status.6)	BOOL	음의 변화율 제한이 잘못되었습니다. 유효하지 않을 경우 명령어는 음의 변화율 감지를 사용하지 않도록 설정하는 ROCNegLimit = 0.0 을 사용합니다.
ROCPeriodInv (Status.7)	BOOL	변화율 기간이 잘못되었습니다. 유효하지 않을 경우 명령어는 변화율 감지를 사용하지 않도록 설정하는 ROCPeriod = 0.0 을 사용합니다.
Overflow (Status.8)	BOOL	오버플로 상태가 감지된 경우 오버플로 비트가 참으로 설정됩니다. 오버플로 상태가 해결된 경우 오버플로 비트가 거짓으로 해제됩니다. CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

OperShelve 태그에 단추 연결

알람 명령어는 원치 않는 알람 재보류를 방지하기 위해 해제에서 설정으로 전환될 경우에만 OperShelve 태그만 처리합니다. ProgUnshelve 태그가 설정되어 있는 동안 작업자가 누름 단추를 눌러 알람을 보류할 경우 ProgUnshelve 태그가 우선 적용되기 때문에 알람이 보류되지 않습니다. ProgUnshelve 가 해제된 후 작업자가 푸시 단추를 놓았다가 다시 누르면 알람을 보류할 수 있습니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570 컨트롤러	예

마이너 폴트 발생 조건:	폴트 유형	폴트 코드
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570 컨트롤러의 경우에만 입력 값이 INF 또는 NAN 입니다.	4	4

연산 상태 플래그를 참조하십시오.

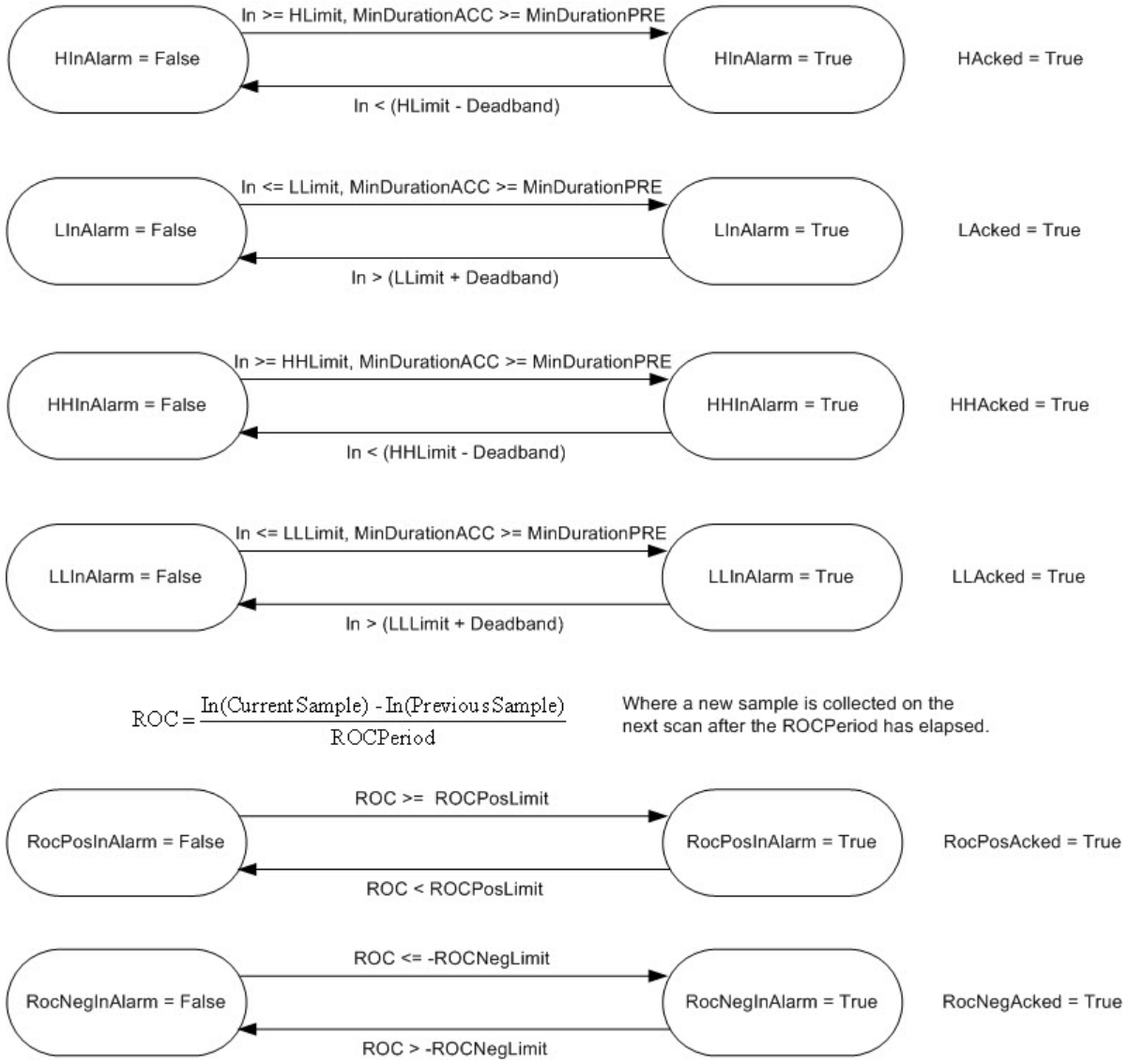
메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

아날로그 알람 상태 다이어그램

다음 도표는 변화하는 알람 상태 및 작업자 명령에 대한 아날로그 알람의 반응 방식을 보여줍니다.

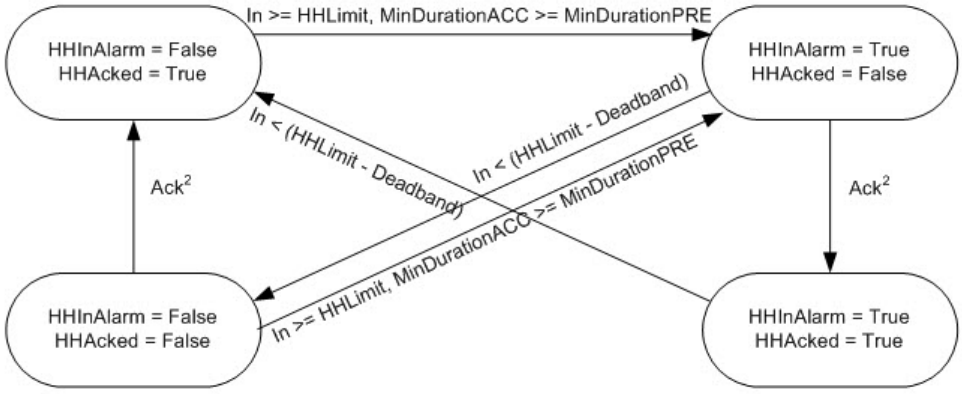
AckRequired = False



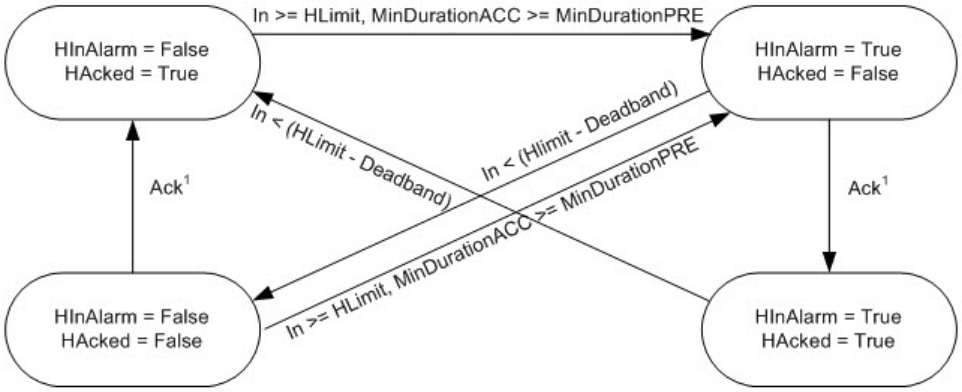
$$ROC = \frac{\ln(\text{Current Sample}) - \ln(\text{Previous Sample})}{\text{ROCPeriod}}$$

Where a new sample is collected on the next scan after the ROCPeriod has elapsed.

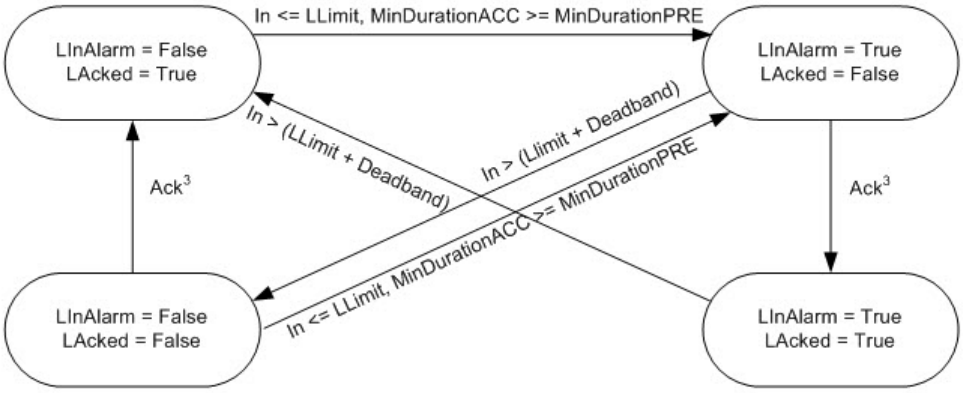
AckRequired = True



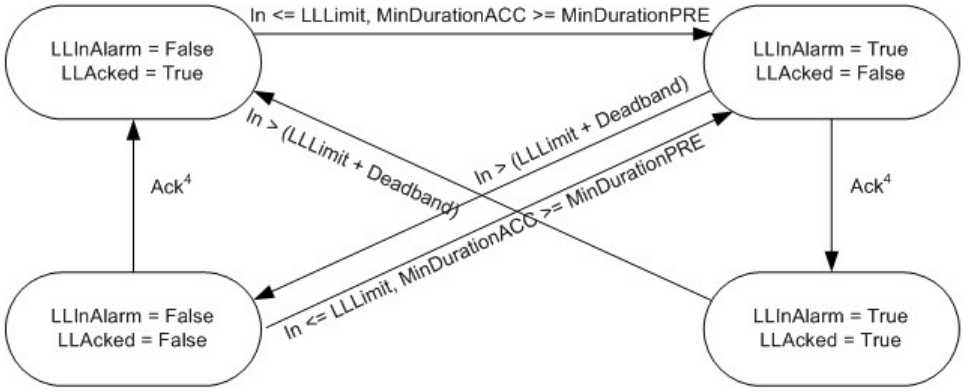
[2] HH alarm condition can be acked by several different ways: HHProgAck, HHOperAck, ProgAckAll, OperAckAll, clients (RSLogix 5000, RSview)



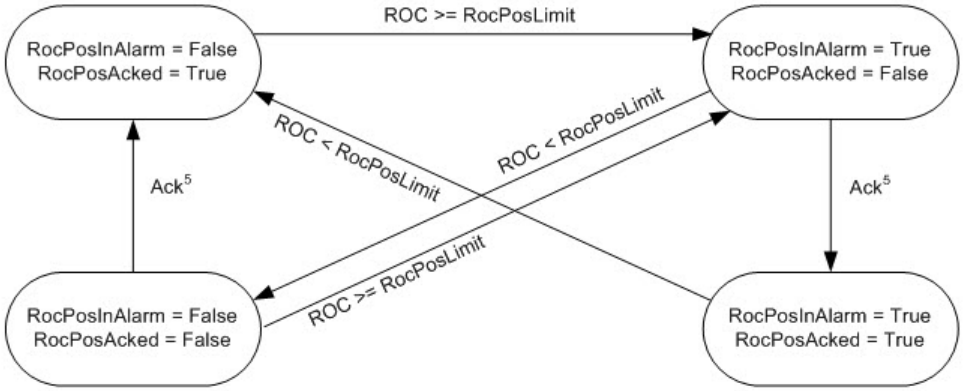
[1] H alarm condition can be acked by several different ways: HProgAck, HOperAck, ProgAckAll, OperAckAll, clients (RSLogix 5000, RSview)



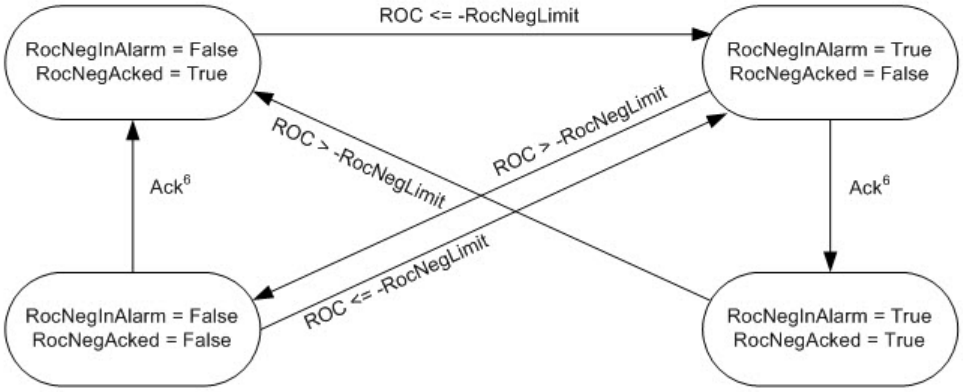
[3] L alarm condition can be acked by several different ways: LProgAck, LOperAck, ProgAckAll, OperAckAll, clients (RSLogix 5000, RSview)



[4] LL alarm condition can be acked by several different ways: LLProgAck, LLOperAck, ProgAckAll, OperAckAll, clients (RSLogix 5000, RSview)



[5] RocPos alarm condition can be acked by several different ways: RocPosProgAck, RocPosOperAck, ProgAckAll, OperAckAll, clients (RSLogix 5000, RSview)

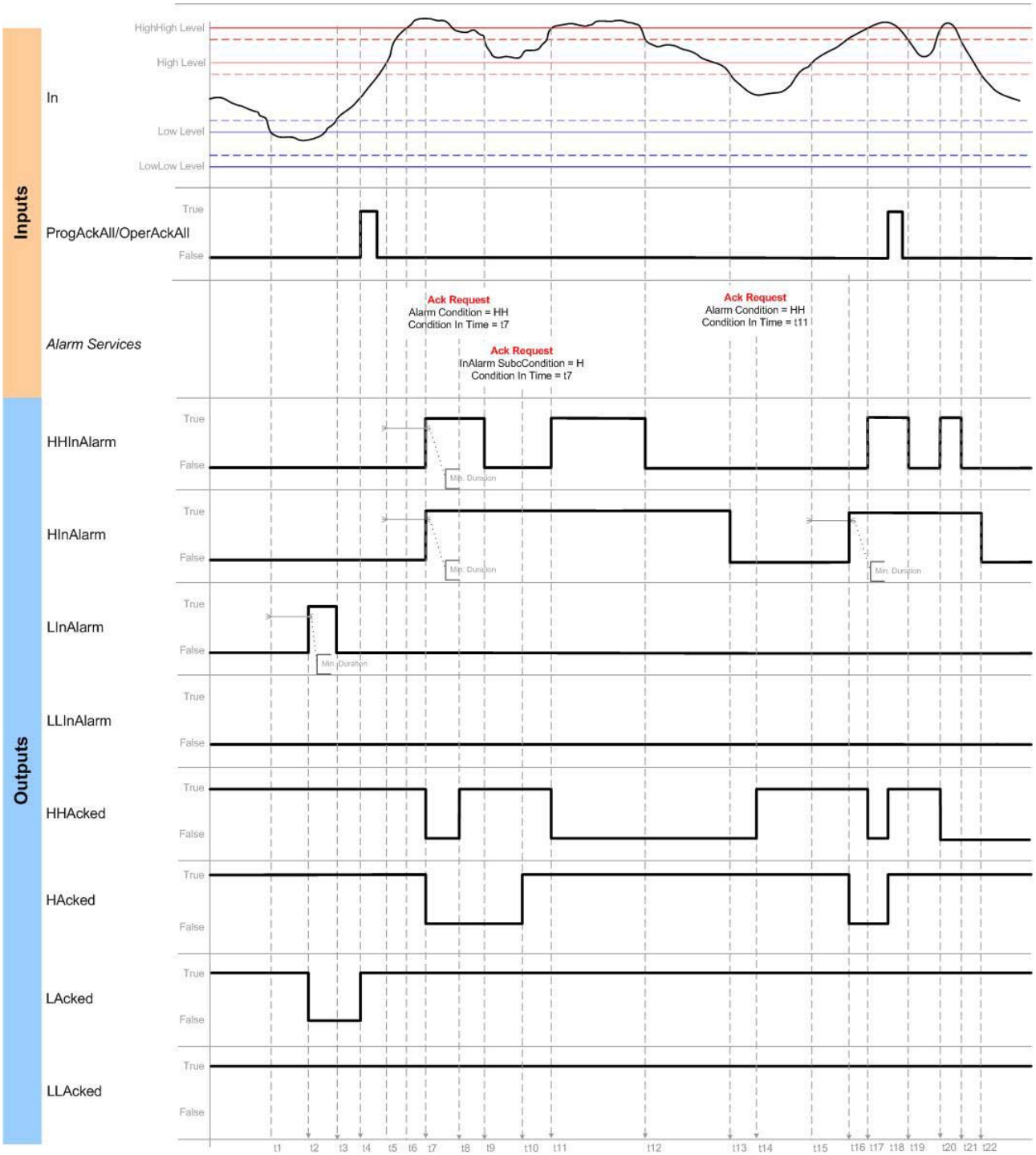


[6] RocNeg alarm condition can be acked by several different ways: RocNegProgAck, RocNegOperAck, ProgAckAll, OperAckAll, clients (RSLogix 5000, RSview)

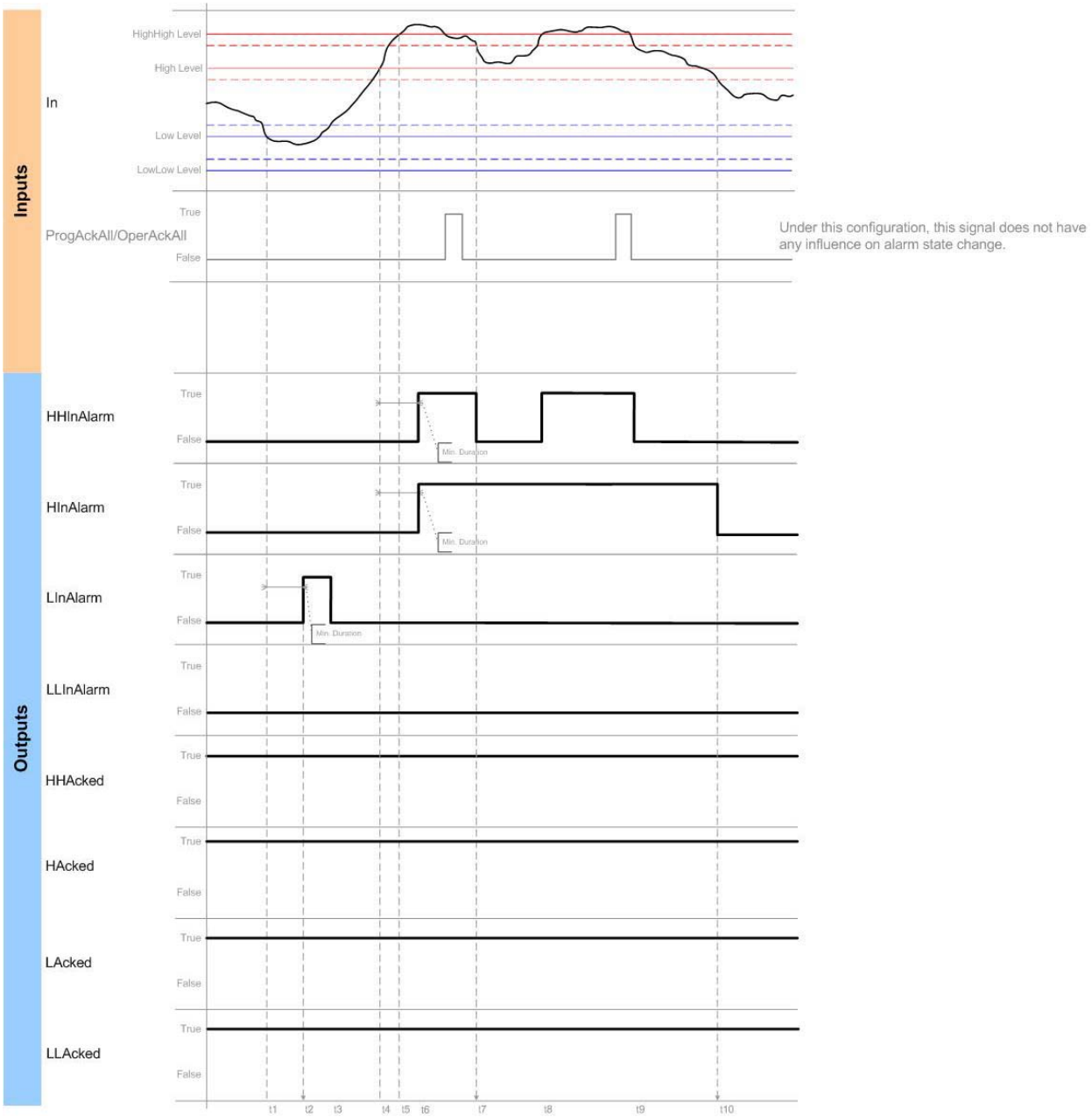
아날로그 알람 타이밍 다이어그램

아래 타이밍 다이어그램은 아날로그 알람의 작동 시퀀스를 보여줍니다.

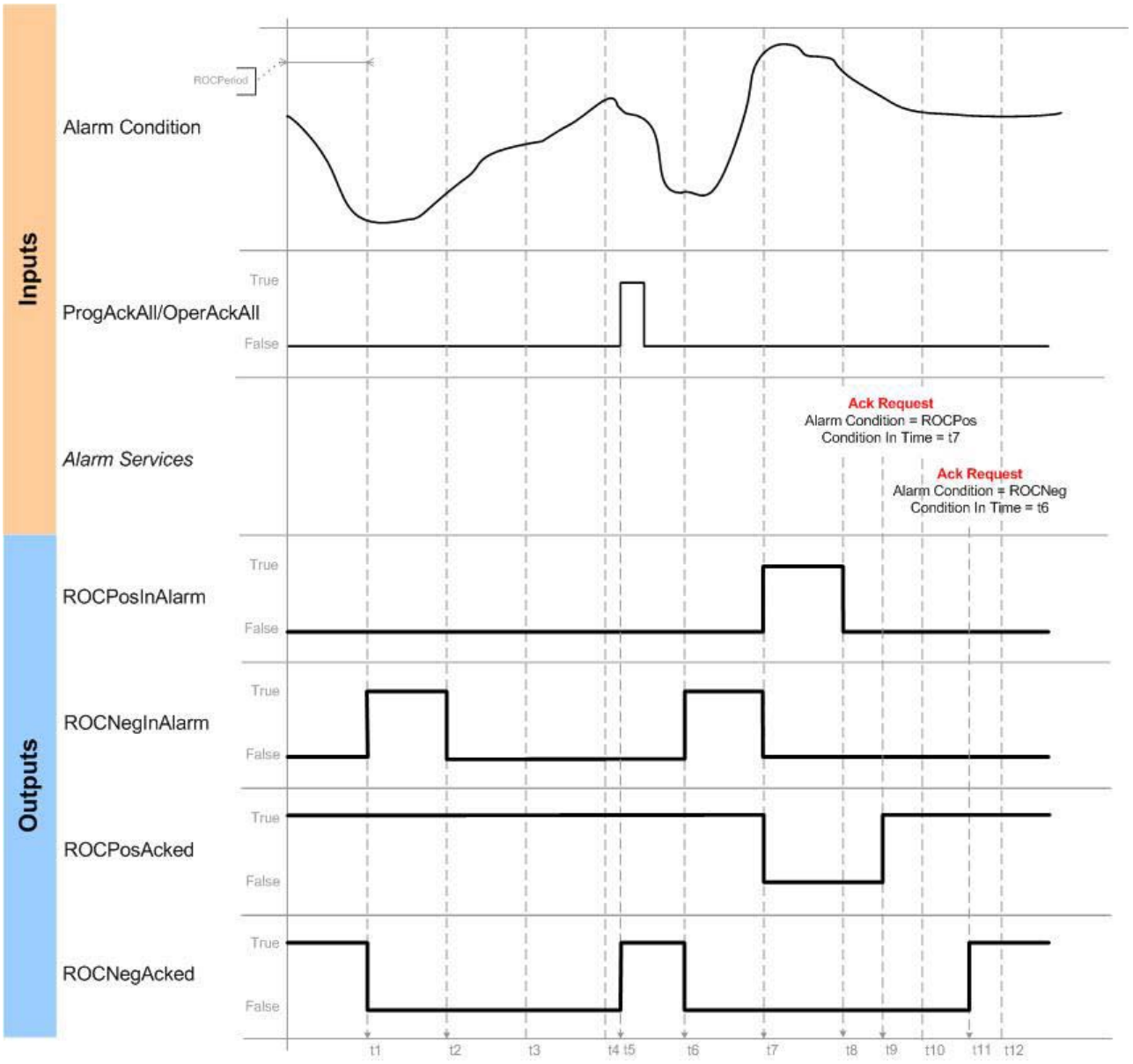
레벨 상태 동작 확인


Inputs
Outputs
In
HighHigh Level
High Level
Low Level
LowLow Level
ProgAckAll/OperAckAll
True
False
Alarm Services
Ack Request
Alarm Condition = HH
Condition In Time = t7
Ack Request
InAlarm SubcCondition = H
Condition In Time = t7
Ack Request
Alarm Condition = HH
Condition In Time = t11
HHInAlarm
HInAlarm
LInAlarm
LLInAlarm
HHAcked
HAcked
LAcked
LLAcked
Min. Duration
t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12 t13 t14 t15 t16 t17 t18 t19 t20 t21 t22

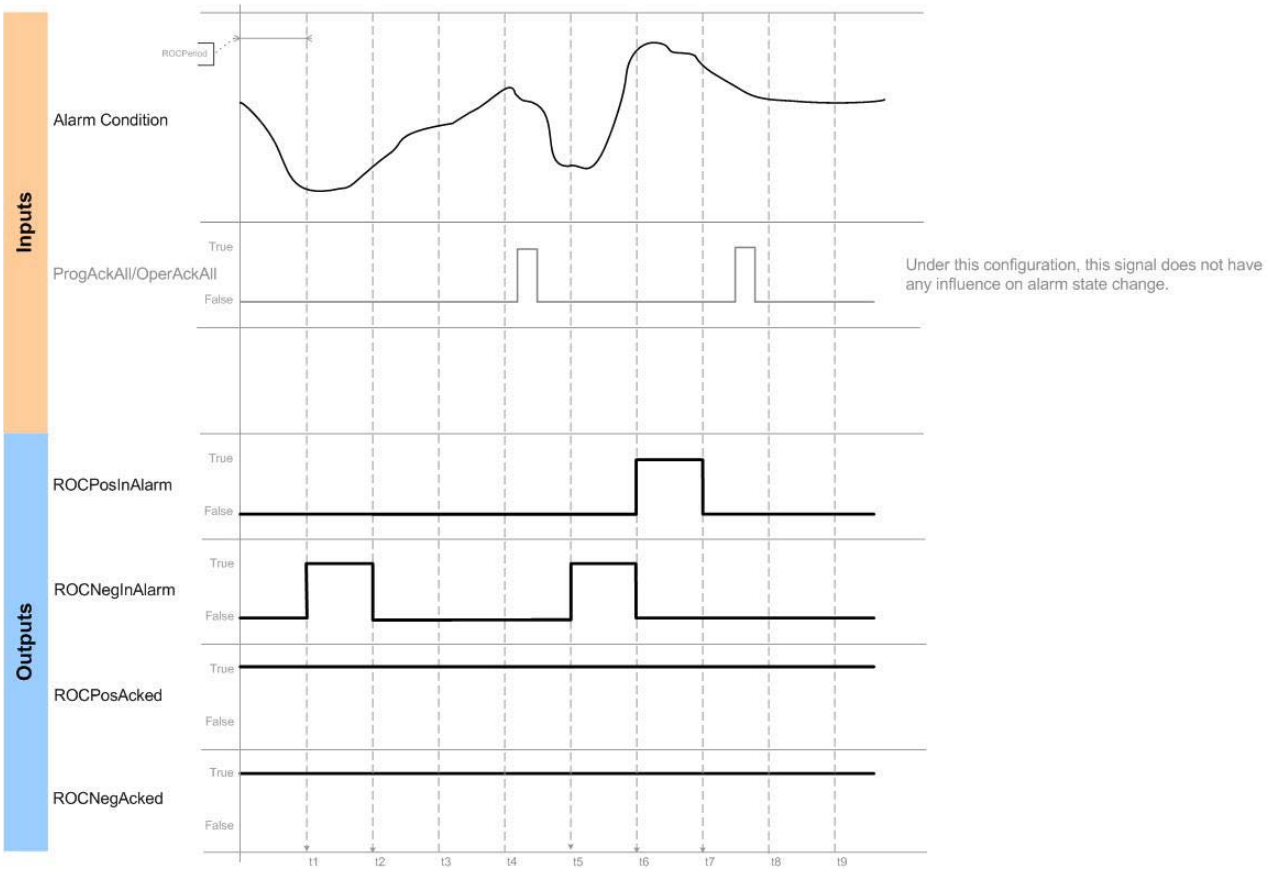
레벨 상태 동작 확인 안 함


Inputs
In
HighHigh Level
High Level
Low Level
LowLow Level
ProgAckAll/OperAckAll
True
False
Under this configuration, this signal does not have any influence on alarm state change.
Outputs
HHInAlarm
HInAlarm
LInAlarm
LLInAlarm
HHAcked
HAcked
LAcked
LLAcked
Min. Duration
t1
t2
t3
t4
t5
t6
t7
t8
t9
t10

ROC 상태 동작 확인


Inputs
Outputs
ROCPeriod
Alarm Condition
True
ProgAckAll/OperAckAll
False
Alarm Services
Ack Request
Alarm Condition = ROCPos
Condition In Time = t7
Ack Request
Alarm Condition = ROCNeg
Condition In Time = t6
ROCPosInAlarm
ROCNegInAlarm
ROCPosAcked
ROCNegAcked
t1
t2
t3
t4 t5
t6
t7
t8
t9
t10
t11
t12

ROC 상태 동작 확인 안 함



실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	렁-출력-조건이 거짓으로 해제됩니다. 모든 ALMA 구조 파라미터가 해제됩니다. 모든 알람 조건이 확인됩니다. 모든 작업자 요청이 해제됩니다. 타임스탬프가 모두 지워집니다. 전달 플래그가 모두 지워집니다.
렁-입력-조건이 거짓임	렁-출력-조건이 거짓으로 해제됩니다.
렁-입력-조건이 참임	렁-출력-조건이 참으로 설정됩니다. 명령어가 실행됩니다.
사후 스캔	렁-출력-조건이 거짓으로 해제됩니다.

평션 블록

조건/상태	취해진 조치
사전 스캔	Tag.EnableOut 이 거짓으로 해제됩니다. 모든 ALMA 구조 파라미터가 해제됩니다. 모든 알람 조건이 확인됩니다. 모든 작업자 요청이 해제됩니다. 타임스탬프가 모두 지워집니다. 전달 플래그가 모두 지워집니다.
Tag.EnableIn 이 거짓임	Tag.EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	명령어가 실행됩니다. Tag.EnableOut 이 참으로 설정됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	Tag.EnableOut 이 거짓으로 해제됩니다.

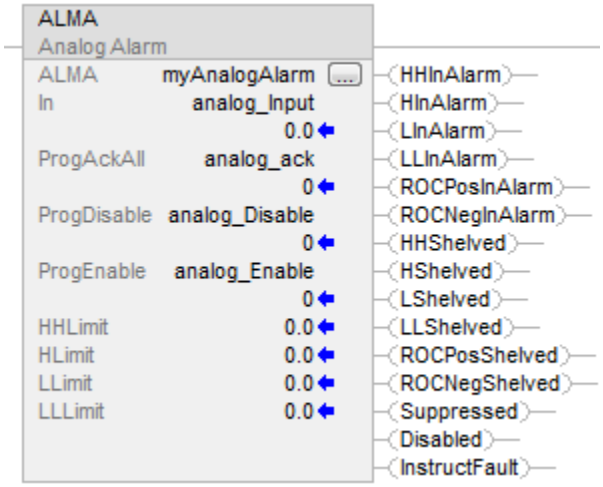
ST(스트럭처드 텍스트)

ST(스트럭처드 텍스트)에서 정상 스캔 중에 EnableIn 은 항상 참입니다. 따라서 명령어가 로직에서 활성화한 제어 경로에 있는 경우 해당 명령어가 실행됩니다.

조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 참조하십시오.

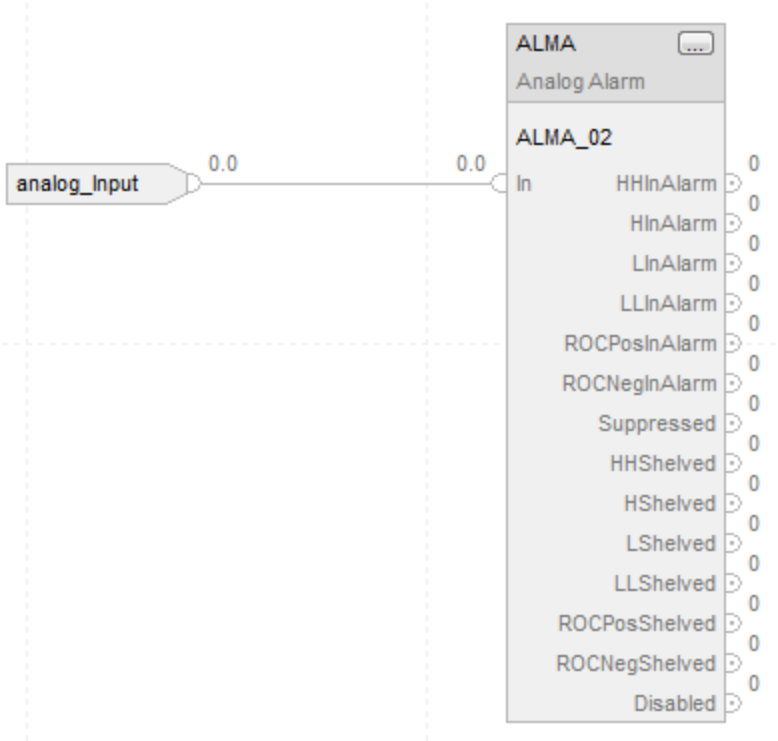
예제

래더 다이어그램



펑션 블록

펑션 블록에 사용된 ALMA 명령어의 예가 아래에 나와 있습니다. 이 예에서는 Tank 32 레벨 트랜스미터(Tank32LT)의 알람 조건이 모니터링됩니다. Tank32LevelAck 태그를 사용하여 이 알람의 모든 조건을 확인합니다.



ST(스트럭쳐드 텍스트)

이 예에서는 Tank 32 레벨 트랜스미터(Tank32LT)의 알람 조건이 모니터링됩니다. Tank32LevelAck 태그를 사용하여 이 알람의 모든 조건을 확인합니다.

```
ALMA(Tank32Level,Tank32LT,Tank32LevelAck,0, 0);
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

연산 상태 플래그 페이지의 963

배열을 통한 인덱스 페이지의 978

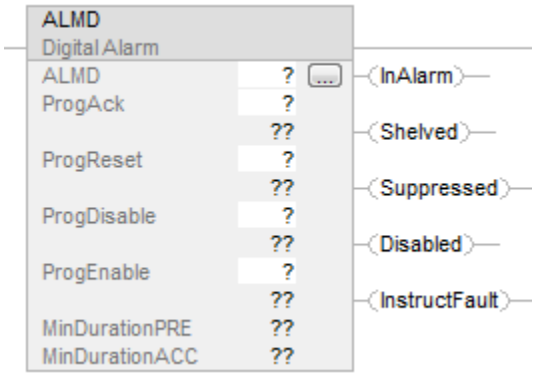
디지털 알람(ALMD)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

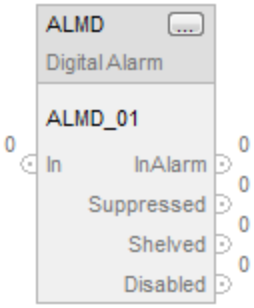
ALMD 명령어는 이산 부울 값에 대한 알람 기능을 제공합니다.

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

```
ALMD (ALMD,In,ProgAck,ProgReset,ProgDisable,ProgEnable)
```

피연산자

래더 다이어그램

피연산자	유형	형식	설명
ALMD tag	ALARM_DIGITAL	구조	ALMD 구조
ProgAck	BOOL	태그 즉시	거짓에서 참으로 전환 후 알람이 확인됩니다(확인이 필요한 경우).
ProgReset	BOOL	태그 즉시	거짓에서 참으로 전환 후 알람이 리셋됩니다(리셋이 필요한 경우).
ProgDisable	BOOL	태그 즉시	참일 경우 알람이 비활성화됩니다(활성화 명령을 오버라이드하지 않음).
ProgEnable	BOOL	태그 즉시	참일 경우 알람이 활성화됩니다(비활성화 명령보다 우선 적용됨).
MinDurationPRE	DINT	즉시	알람 조건이 보고되기까지 지속되어야 하는 시간(밀리초)을 지정합니다.
MinDurationACC	DINT	즉시	알람의 MinDuration 타이머에 해당하는 현재 누적기 값을 나타냅니다.

평션 블록

피연산자	유형	형식	설명
ALMD tag	ALARM_DIGITAL	구조	ALMD 구조

ST(스트럭쳐드 텍스트)

피연산자	유형	형식	설명
ALMD tag	ALARM_DIGITAL	구조	ALMD 구조
ProgAck	BOOL	태그 즉시	거짓에서 참으로 전환 후 알람이 확인됩니다(확인이 필요한 경우).
ProgReset	BOOL	태그 즉시	거짓에서 참으로 전환 후 알람이 리셋됩니다(리셋이 필요한 경우).
ProgDisable	BOOL	태그 즉시	참일 경우 알람이 비활성화됩니다(활성화 명령을 오버라이드하지 않음).
ProgEnable	BOOL	태그 즉시	참일 경우 알람이 활성화됩니다(비활성화 명령보다 우선 적용됨).
MinDurationPRE	DINT	즉시	알람 조건이 보고되기까지 지속되어야 하는 시간(밀리초)을 지정합니다.
MinDurationACC	DINT	즉시	알람의 MinDuration 타이머에 해당하는 현재 누적기 값을 나타냅니다.

ST(스트럭쳐드 텍스트) 내 식의 구문에 대한 자세한 내용은 ST(스트럭쳐드 텍스트) 구문을 참조하십시오.

ALMD 구조

입력 파라미터

입력 파라미터	데이터 유형	설명
활성화 입력	BOOL	래더 다이어그램: 렁 상태에 해당합니다. 처리에 영향을 주지 않습니다. 펑션 블록: 거짓으로 해제되면 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 설정하면 명령어가 실행됩니다. 기본값은 참입니다. ST(스트럭쳐드 텍스트): 영향을 받지 않습니다. 명령어가 항상 실행됩니다.

입력 파라미터	데이터 유형	설명
In	BOOL	명령어에 대한 디지털 신호 입력. 기본값은 거짓입니다. 래더 다이어그램: 렁 조건을 따릅니다. 렁 조건이 참인 경우 참으로 설정됩니다. 렁 조건이 거짓인 경우 거짓으로 해제됩니다. ST(스트럭쳐드 텍스트): 명령어 피연산자에서 복사됩니다.
InFault	BOOL	입력에 불량 상태 표시기. 사용자 응용 프로그램에서 InFault 가 입력 신호에 에러가 있음을 나타내도록 설정할 수 있습니다. 설정된 경우 명령어는 InFaulted (Status.1)를 설정하고, 거짓으로 해제된 경우 명령어에 의해 InFaulted (Status.1)가 거짓으로 해제됩니다. 어느 경우이든 명령어는 In 에서 알람 조건을 계속 평가합니다. 기본값은 거짓(올바른 상태)입니다.
Condition	BOOL	알람 활성화 방법을 지정합니다. Condition 이 참으로 설정되면 In 이 참으로 설정된 경우 알람 조건이 활성화되고, Condition 이 거짓으로 해제되면 In 이 거짓으로 해제된 경우 알람 조건이 활성화됩니다. 기본값은 참입니다.
AckRequired	BOOL	알람 확인이 필요한지 여부를 지정합니다. 참으로 설정된 경우 확인이 필요합니다. 거짓으로 해제된 경우 확인이 필요하지 않으며 Acked 가 항상 참으로 설정됩니다. 기본값은 참입니다.
Latched	BOOL	알람을 래치할지 여부를 지정합니다. 초기화 명령이 수신될 때까지 래치된 알람은 알람 조건이 거짓이 될 때 InAlarm 으로 남아 있습니다. 참으로 설정된 경우 알람이 래치되고, 거짓으로 해제되면 알람이 래치 해제됩니다. 기본값은 거짓입니다. 알람 조건이 거짓인 경우에만 래치된 알람을 초기화할 수 있습니다.
ProgAck	BOOL	알람을 확인하도록 사용자 프로그램에 의해 참으로 설정됩니다. 알람이 확인되지 않은 경우에만 적용됩니다. 거짓 - 참 전환이 필요합니다. 기본값은 거짓입니다. 래더 다이어그램: 명령어 피연산자에서 복사됩니다. ST(스트럭쳐드 텍스트): 명령어 피연산자에서 복사됩니다.

입력 파라미터	데이터 유형	설명
OperAck	BOOL	알람을 확인하려면 작업자 인터페이스에서 참으로 설정합니다. 알람이 확인되지 않은 경우에만 적용됩니다. 명령어가 이 파라미터를 지웁니다. 기본값은 거짓입니다.
ProgReset	BOOL	래치된 알람을 리셋하도록 사용자 프로그램에 의해 참으로 설정됩니다. 래치된 알람이 InAlarm 이고 알람 조건이 거짓인 경우에만 적용됩니다. 거짓 - 참 전환이 필요합니다. 기본값은 거짓입니다. 래더 다이어그램: 명령어 피연산자에서 복사됩니다. ST(스트럭쳐드 텍스트): 명령어 피연산자에서 복사됩니다.
OperReset	BOOL	래치된 알람을 리셋하기 위해 작업자 인터페이스에서 참으로 설정합니다. 래치된 알람이 InAlarm 이고 알람 조건이 거짓인 경우에만 적용됩니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
ProgSuppress	BOOL	알람을 억제하도록 사용자 프로그램에 의해 참으로 설정됩니다. 기본값은 거짓입니다.
OperSuppress	BOOL	알람을 억제하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
ProgUnsuppress	BOOL	알람을 억제 해제하도록 사용자 프로그램에 의해 참으로 설정됩니다. 억제 명령보다 우선 순위를 갖습니다. 기본값은 거짓입니다.
OperUnsuppress	BOOL	알람을 억제 해제하기 위해 작업자 인터페이스에서 참으로 설정합니다. 억제 명령보다 우선 순위를 갖습니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.

입력 파라미터	데이터 유형	설명
OperShelve	BOOL	알람을 보류 또는 재보류하려면 작업자 인터페이스에서 이 파라미터를 참으로 설정합니다. 한 프로그램 스캔은 거짓, 다음 프로그램 스캔은 참으로 전환되어야 합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다. 보류 해제 명령이 보류 명령보다 우선합니다. 알람을 보류하면 알람 처리가 연기됩니다. 보류는 시간이 제한된다는 점만 제외하고 알람 억제와 유사합니다. 알람이 보류된 동안 확인될 경우 알람이 다시 활성화되더라도 계속 확인되어 있습니다. 해당 시점에 알람이 계속 활성화되어 있는 경우 보류 기간이 종료되면 알람이 확인되지 않습니다.
ProgUnshelve	BOOL	알람을 보류 해제하도록 사용자 프로그램에 의해 참으로 설정됩니다. 보류 명령보다 우선 순위를 갖습니다. 기본값은 거짓입니다. 알람 보류에 대한 자세한 내용은 OperShelve 파라미터 설명을 참조하십시오.
OperUnshelve	BOOL	알람을 보류 해제하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 보류 명령보다 우선 순위를 갖습니다. 기본값은 해제 상태입니다. 알람 보류에 대한 자세한 내용은 OperShelve 파라미터 설명을 참조하십시오.
ProgDisable	BOOL	알람을 비활성화하도록 사용자 프로그램에 의해 참으로 설정됩니다. 기본값은 거짓입니다. 래더 다이어그램: 명령어 피연산자에서 복사됩니다. ST(스트럭쳐드 텍스트): 명령어 피연산자에서 복사됩니다.
OperDisable	BOOL	알람을 비활성화하려면 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 참으로 해제됩니다. 기본값은 거짓입니다.

입력 파라미터	데이터 유형	설명
ProgEnable	BOOL	알람을 활성화하도록 사용자 프로그램에 의해 참으로 설정됩니다. 사용 안 함 명령보다 우선 순위를 갖습니다. 기본값은 거짓입니다. 래더 다이어그램: 명령어 피연산자에서 복사됩니다. ST(스트럭쳐드 텍스트): 명령어 피연산자에서 복사됩니다.
OperEnable	BOOL	알람을 활성화하려면 작업자 인터페이스에서 이 파라미터를 참으로 설정합니다. 사용 안 함 명령보다 우선 순위를 갖습니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
AlarmCountReset	BOOL	알람 카운트를 0 으로 리셋하기 위해 작업자 인터페이스에서 참으로 설정합니다. 알람 명령어에 의해 이 파라미터가 거짓으로 해제됩니다. 기본값은 거짓입니다.
UseProgTime	BOOL	컨트롤러의 클럭을 사용하여 알람 상태 변경 이벤트에 타임스탬프를 찍을지 또는 ProgTime 값을 타임스탬프를 찍을지를 지정합니다. 참으로 설정된 경우 ProgTime 값으로 타임스탬프가 인쇄되고, 거짓으로 해제된 경우 컨트롤러의 클럭을 기준으로 타임스탬프가 인쇄됩니다. 기본값은 거짓입니다.
ProgTime	LINT	UseProgTime 이 참으로 설정된 경우 이 값은 모든 이벤트에 대한 타임스탬프 값으로 사용됩니다. 이 경우 응용 프로그램은 이벤트 시퀀스 입력 모듈과 같은 알람 소스에서 가져온 타임스탬프를 적용할 수 있습니다.
Severity	DINT	알람의 심각도. 이는 컨트롤러의 알람 처리에 영향을 주지는 않지만 알람 구독자 측에서 함수를 정렬 및 필터링하는 데 사용할 수 있습니다. 유효값 = 1 ~ 1000(1000 = 가장 심각함; 1 = 가장 심각하지 않음) 기본값 = 500.
MinDurationPRE	DINT	알람이 InAlarm 으로 표시되고 클라이언트에 알람 알림이 전송되기 전에 알람 조건이 참으로 유지되어야 하는 미리 설정된 최소 지속 시간(밀리초). 컨트롤러는 알람 조건이 감지되는 즉시 알람 데이터를 수집하므로 최소 기간에 도달할 때까지 기다리는 동안 데이터가 손실되지 않습니다. 유효값 = 0 ~ 2147483647 기본값 = 0.

입력 파라미터	데이터 유형	설명
ShelveDuration	DINT	알람을 보류할 기간(분). 알람을 보류하면 알람 처리가 연기됩니다. 보류는 시간이 제한된다는 점만 제외하고 알람 억제와 유사합니다. 알람이 보류된 동안 확인될 경우 알람이 다시 활성화되더라도 계속 확인되어 있습니다. 해당 시점에 알람이 계속 활성화되어 있는 경우 보류 기간이 종료되면 알람이 확인되지 않습니다. 최소 시간은 1 분입니다. 최대 시간은 MaxShelveDuration 으로 정의됩니다.
MaxShelveDuration	DINT	알람을 보류할 수 있는 최대 기간(분). 알람 보류에 대한 자세한 내용은 ShelveDuration 파라미터 설명을 참조하십시오.

출력 파라미터

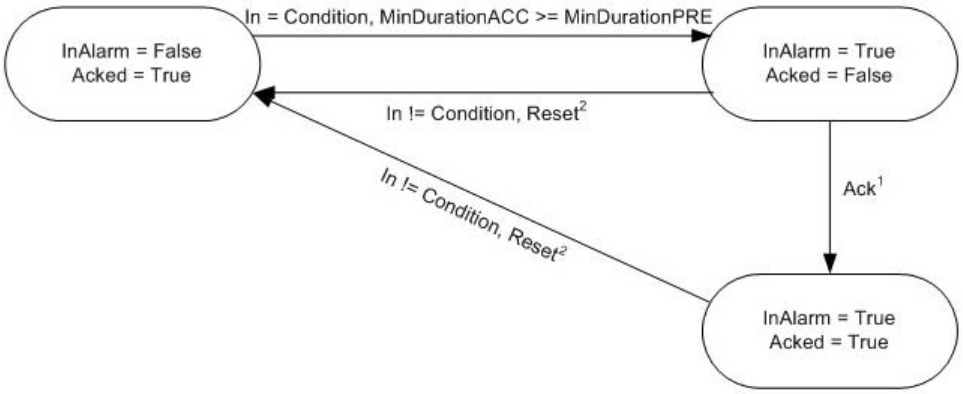
출력 파라미터	데이터 유형	설명
EnableOut	BOOL	출력 활성화.
InAlarm	BOOL	알람 활성 상태. 알람이 활성인 경우 참으로 설정되고, 알람이 활성이 아니면(정상 상태) 거짓으로 해제됩니다.
Acked	BOOL	알람이 확인된 상태. 알람이 확인된 경우 참으로 설정되고, 알람이 확인되지 않은 경우 거짓으로 해제됩니다. AckRequired 가 거짓으로 해제된 경우 Acked 는 항상 참으로 설정됩니다.
InAlarmUnack	BOOL	알람이 활성화되고 확인된 상태. 알람이 활성(InAlarm 이 참)이고 확인되지 않은 경우(Acked 가 거짓임) 참으로 설정되고, 알람이 비활성이고 확인되었거나 둘 다에 해당하는 경우 거짓으로 해제됩니다.
Suppressed	BOOL	알람 억제 상태. 알람이 억제된 경우 참으로 설정되고, 알람이 억제되지 않은 경우 거짓으로 해제됩니다.
Shelved	BOOL	알람의 보류 상태. 알람이 보류된 경우 참으로 설정되고, 알람이 보류 해제된 경우 거짓으로 해제됩니다. 알람을 보류하면 알람 처리가 연기됩니다. 보류는 시간이 제한된다는 점만 제외하고 알람 억제와 유사합니다. 알람이 보류된 동안 확인될 경우 알람이 다시 활성화되더라도 계속 확인되어 있습니다. 보류 기간이 종료되면 알람이 확인되지 않은 상태가 됩니다.
Disabled	BOOL	알람 사용 안 함 상태. 알람이 활성화되지 않은 경우 참으로 설정됩니다. 알람이 활성화되면 거짓으로 해제됩니다.
Commissioned	BOOL	알람의 의뢰됨 상태. 알람이 의뢰된 경우 참으로 설정되고, 알람 의뢰를 해제하면 거짓으로 해제됩니다. 현재 항상 참으로 설정됩니다.

MinDurationACC	DINT	사용하지 않습니다. 값이 항상 0 입니다.
AlarmCount	DINT	알람이 활성화(InAlarm 설정)된 횟수. 최고 값에 도달하면 카운터의 값이 최고 카운트 값으로 유지됩니다.
InAlarmTime	LINT	알람 감지의 타임스탬프
AckTime	LINT	알람 확인의 타임스탬프. 알람에 확인이 필요하지 않은 경우 이 타임스탬프는 알람 시간과 같습니다.
RetToNormalTime	LINT	정상 상태로 돌아가는 알람의 타임스탬프
AlarmCountReset Time	LINT	알람 카운트가 초기화된 때를 나타내는 타임스탬프.
ShelveTime	LINT	알람이 마지막으로 보류된 시간을 나타내는 타임스탬프. 이 값은 알람이 보류되면 컨트롤러에 의해 설정됩니다. 알람은 여러 번 보류 및 보류 해제할 수 있습니다. 알람을 보류할 때마다 타임스탬프가 현재 시간으로 설정됩니다. 알람 보류에 대한 자세한 내용은 Shelved 파라미터 설명을 참조하십시오.
UnshelveTime	LIN	알람이 보류 해제될 시간을 나타내는 타임스탬프. 알람이 보류될 때마다 이 값이 설정됩니다(알람이 이미 보류된 경우에도). ShelveDuration 을 현재 시간에 추가하여 타임스탬프가 결정됩니다. 프로그래밍 방식으로 또는 연산자가 알람을 보류 해제할 경우 값은 현재 시간으로 설정됩니다. 알람 보류에 대한 자세한 내용은 Shelved 파라미터 설명을 참조하십시오.
Status	DINT	혼합된 상태 표시: Status.0 = InstructFault Status.1= InFaulted Status.2 = SeverityInv
InstructFault (Status.0)	BOOL	명령어 에러 조건이 있습니다. 마이너 또는 메이저 컨트롤러 에러가 아닙니다. 나머지 상태 비트를 확인하여 발생한 에러를 판단합니다.
InFaulted (Status.1)	BOOL	사용자 프로그램에서 InFault 가 품질이 좋지 않은 입력 데이터를 나타내도록 설정했습니다. 알람은 알람 조건에 대해 In 평가를 계속합니다.
SeverityInv (Status.2)	BOOL	알람 심각도 구성. 심각도 < 1 이면 명령어는 Severity = 1 을 사용합니다. 심각도 >1000 이면 명령어는 Severity = 1000 을 사용합니다.

디지털 알람 상태 다이어그램

Acknowledgement Required, Latched

AckRequired = True, Latched = True

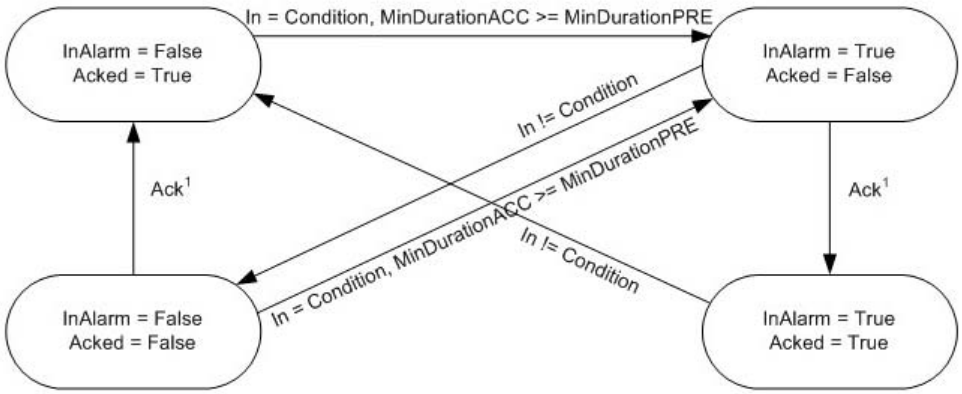


[1] Alarm can be acked by several different ways: ProgAck, OperAck, clients (RSLogix 5000, RSview)

[2] Alarm can be reset by several different ways: ProgReset, OperReset, clients (RSLogix 5000, RSview)

Acknowledgement Required, Not Latched

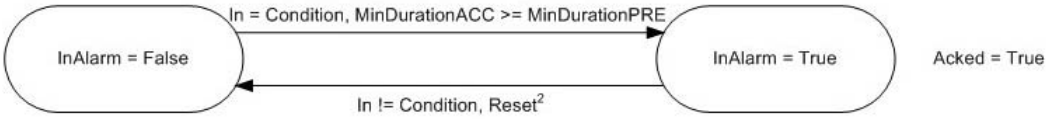
AckRequired = True, Latched = False



[1] Alarm can be acked by several different ways: ProgAck, OperAck, clients (RSLogix 5000, RSview)

Acknowledgement Not Required, Latched

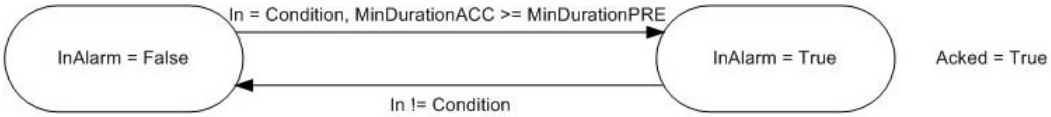
AckRequired = False, Latched = True



[2] Alarm can be reset by several different ways: ProgReset, OperReset, clients (RSLogix 5000, RSview)

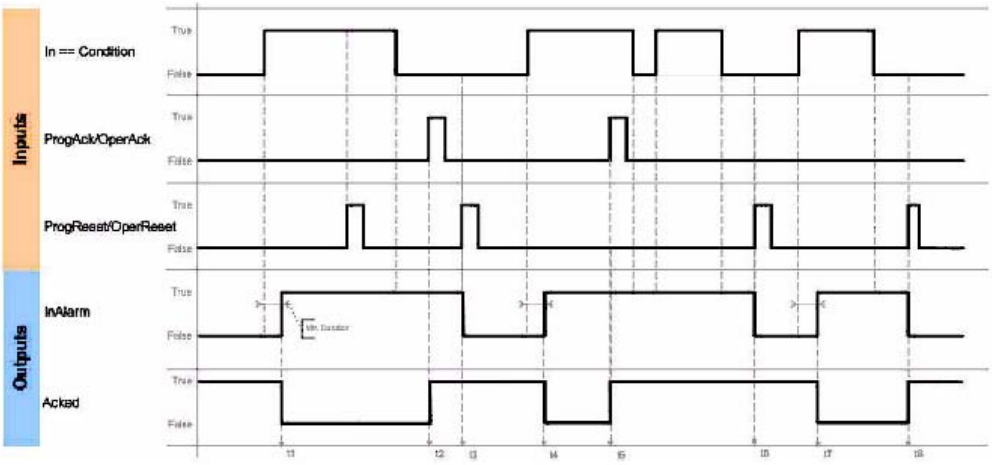
Acknowledgement Not Required, Not Latched

AckRequired = False, Latched = False

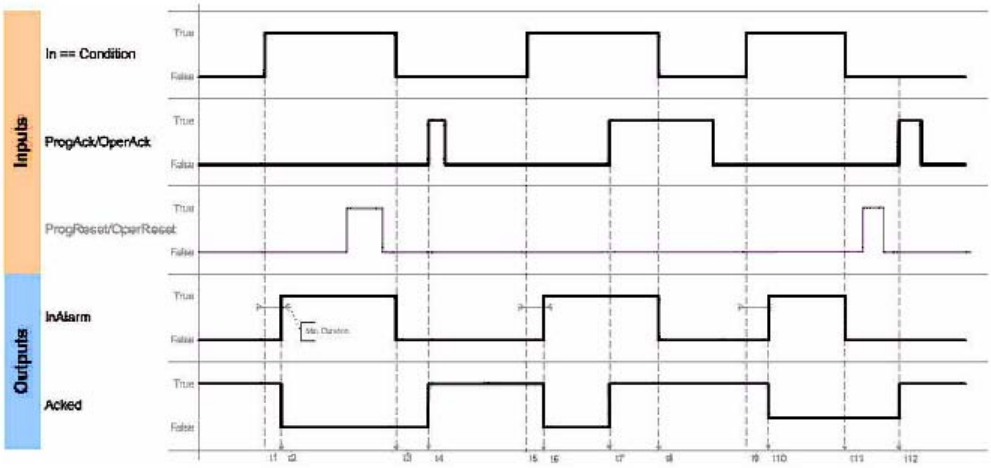


디지털 알람 타이밍 다이어그램

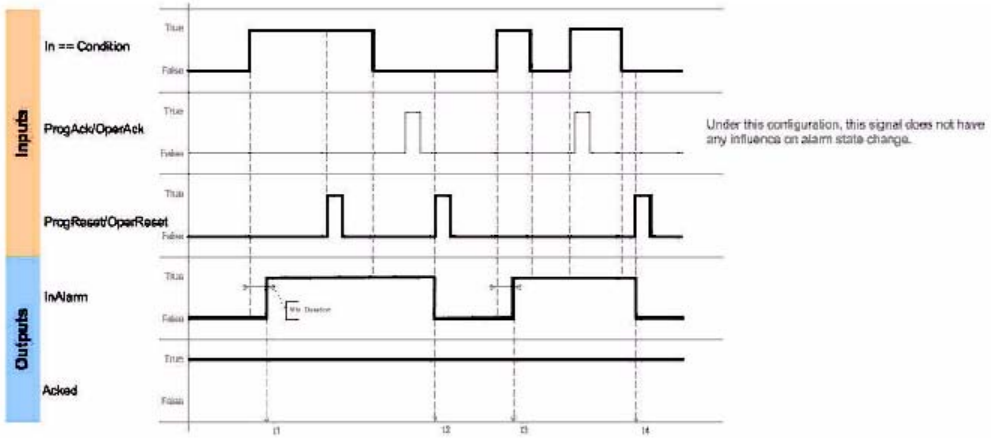
ALMD 알람 확인 필요하고 래치됨



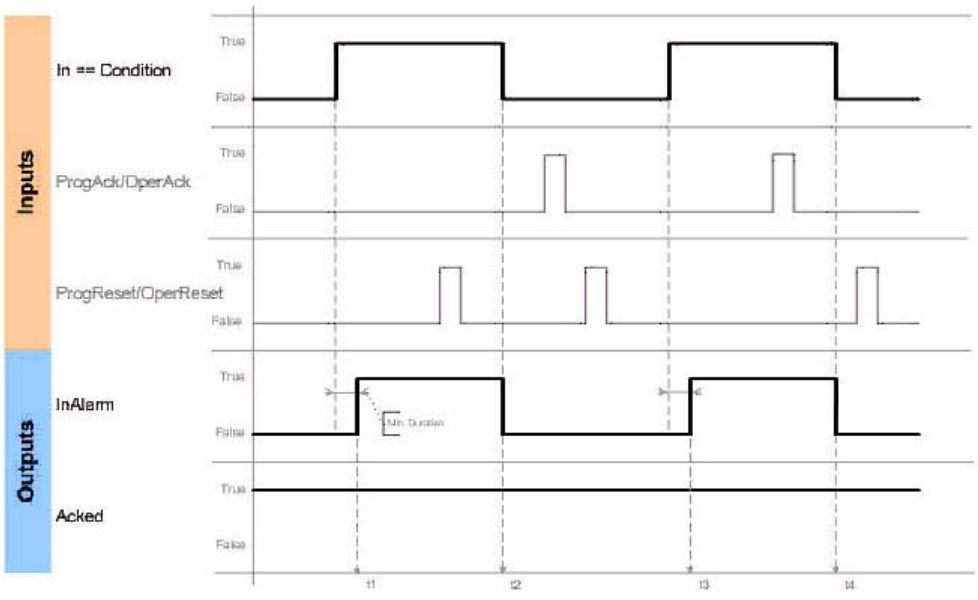
ALMD 알람 확인 필요하지만 래치되지 않음



ALMD 알람 확인이 필요하지 않고 래치됨



ALMD 알람 확인이 필요하지 않고 래치되지 않음



OperShelve 태그에 단추 연결

원치 않는 알람의 재보류를 방지하기 위해 알람 명령어는 한 프로그램 스캔에 이은 다음 스캔에서 거짓에서 참으로 전환된 경우에만 OperShelve 태그를 처리합니다. ProgUnshelve 태그가 참인 상태에서 작업자가 누름 단추를 눌러 알람을 보류하는 경우 ProgUnshelve 태그가 우선 적용되기 때문에 알람이 보류되지 않습니다. 그러나 프로그램 스캔은 밀리초 이내에 완료되기 때문에 ProgUnshelve 태그가 거짓으로 해제되었더라도 다수의 프로그램 스캔 동안 OperShelve 태그가 계속 참으로 되도록 작업자가 누름 단추를 계속 누를 수 있습니다. 이 경우 알람이 보류되지 않습니다.

작업자가 단추를 놓았다가 다시 누르는 식으로 알람을 보류할 수 있습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	EnableOut 이 거짓으로 해제됨 InAlarm 출력이 거짓으로 해제됨 Shelved 출력이 거짓으로 해제됨 Acked 출력이 참으로 설정됩니다. 모든 알람 조건이 확인됩니다. 모든 작업자 요청이 해제됩니다. 타임스탬프가 모두 지워집니다.
렁-입력-조건이 거짓임	렁이 거짓으로 해제됩니다. In 파라미터가 거짓으로 해제됨 명령어가 실행됩니다.
렁-입력-조건이 참임	렁이 참으로 설정됩니다. In 파라미터가 참으로 설정됨 명령어가 실행됩니다.
사후 스캔	렁 비트가 거짓으로 해제됩니다.

평션 블록

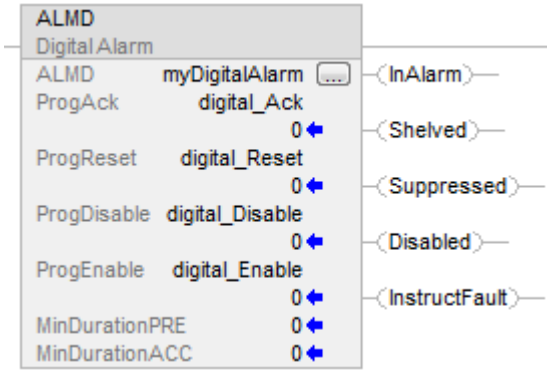
조건/상태	취해진 조치
사전 스캔	Tag.EnableOut 이 거짓으로 해제됩니다. InAlarm 출력이 거짓으로 해제됨 Shelved 출력이 거짓으로 해제됨 Acked 출력이 참으로 설정됩니다. 모든 작업자 요청이 해제됩니다. 타임스탬프가 모두 지워집니다.
Tag.EnableIn 이 거짓임	Tag.EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	명령어가 실행됩니다. Tag.EnableOut 이 참으로 설정됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	Tag.EnableOut 이 거짓으로 해제됩니다.

ST(스트럭처드 텍스트)

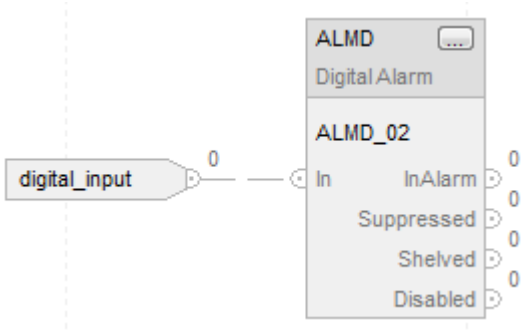
조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 참조하십시오.

예

래더 다이어그램



펑션 블록



ST(스트럭처드 텍스트)

ST(스트럭처드 텍스트)에서 ALMD 명령어의 예가 아래에 나와 있습니다. 이 예에서 두 개의 모터 고장 신호가 결합되어 한 신호가 발생하면 모터 폴트 알람이 활성화됩니다. Motor101Ack 태그를 사용하여 알람을 확인합니다.

```
Motor101FaultConditions := Motor101Overtemp OR Motor101FailToStart;
```

```
ALMD(Motor101Fault,Motor101FaultConditions,Motor101Ack,0,
0,0 );
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

연산 상태 플래그 페이지의 963

배열을 통한 인덱스 페이지의 978

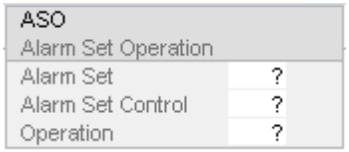
알람 집합 동작(ASO)

이 정보는 Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580 및 GuardLogix 5580 컨트롤러에 적용됩니다.

알람 집합 동작 명령어는 지정된 알람 집합의 모든 알람 조건에 대해 지정된 동작을 지시합니다. 알람 집합 동작 명령어는 지정된 알람 집합의 모든 알람 조건에 대해 알람 동작의 비동기식 실행을 시작하는 데 사용됩니다. 이 명령어는 지정된 알람 집합의 알람 조건을 반복하며 각 조건에 대한 동작 실행을 요청하는 내부 플래그를 설정합니다. 내부 플래그는 기존의 사용자 액세스 가능 Progxxx 비트와 그 목적 및 우선 순위가 동일하며, 지정된 알람 집합의 각 특정 알람 조건에 대한 다음 정기 평가 시 해당 알람 집합의 모든 알람 조건에 대해 처리됩니다.

사용 가능한 언어

래더 다이어그램



함수 블록 다이어그램

이 명령어는 평션 블록 다이어그램에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

ASO (Alarm Set, Alarm Set Control, Operation)

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 동일한 태그(ALARM_SET_CONTROL)를 2 개 이상의 명령어 호출을 위한 파라미터로 사용할 경우.
- 사용자 응용 프로그램에서 .LastState 구조 구성원을 수정할 경우.

주의: 알람 집합 제어 구조에는 내부 상태 정보가 포함되어 있습니다. 실행 모드 중 구성 피연산자 중 하나가 변경되면 보류 중인 편집을 수락하고 컨트롤러 모드를 프로그램 모드에서 실행 모드로 전환하여 변경 내용을 적용합니다.

다음 표에는 명령어를 구성하는 데 사용되는 피연산자가 나와 있습니다.

피연산자	데이터 유형	형식	설명
Alarm Set	ALARM_SET	AlarmSet	ALARM_SET 구조는 이 명령어로 인해 작동되는 알람 조건을 나타냅니다.
Alarm Set Control	ALARM_SET_CONTROL	태그	이 데이터 유형은 3 개의 BOOL 플래그를 포함합니다. • EnableIn • EnableOut • LastState 이 명령어는 레벨이 아닌 에지(.EnableIn 이 거짓에서 참으로 전환됨)에 반응합니다. EnableOut 은 항상 .EnableIn 으로 설정됩니다. 명령어 연산 수행에 대한 요청은 ProgXXX 플래그와 우선 순위가 동일합니다.
연산		immediate	이 피연산자는 목록에서 선택하거나 정수 값으로 입력할 수 있습니다. 0 - 확인 1 - 리셋 2 - 활성화 3 - 비활성화 4- 보류 해제 5 - 억제 6 - 억제 해제 7 - ResetAlarmCount

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

조건/상태	취해진 조치
사전 스캔	이 명령어는 모든 ALARM_SET 구조 구성원을 지웁니다.
렁-입력-조건이 거짓임	이 명령어는 .EnableOut 및 .LastState 구조 구성원을 지웁니다.
렁-입력-조건이 참임	.LastState 가 거짓이면 이 명령어가 연산을 시작하고 .LastState 구조 구성원을 참으로 설정합니다. .EnableOut 구조 구성원은 항상 참으로 설정됩니다.
사후 스캔	이 명령어는 모든 ALARM_SET 구조 구성원을 지웁니다.

연산

알람 집합 동작 명령어는 지정된 알람 집합에서 다음 알람 동작 중 하나의 비동기식 실행을 시작합니다.

- 확인
- 리셋
- 활성화
- 비활성활
- 보류 해제
- 억제
- 억제 해제
- ResetAlarmCount

명령어가 지정된 알람 집합 또는 중첩된 알람 집합에 포함된 모든 알람 조건을 반복하여 특정 알람 조건에 필요한 동작 수행 요청을 나타내는 내부 플래그를 설정합니다. 다음을 제외하고 명령어에 의해 반복되는 모든 알람 조건에 대해 이 동작이 시작됩니다.

- 알람 동작을 지원하지 않도록 구성된 알람 조건
- 사용되지 않도록 구성된 알람 조건

명령어에 의해 특정 알람 조건에 대해 알람 동작이 시작된 경우 다음 알람 조건 정기 평가 시 해당 동작이 수행됩니다.

모순되는 알람 동작을 시작하기 위해 동일한 알람 집합에 대해 명령어를 여러 번 호출할 경우 마지막으로 요청된 동작이 항상 알람 집합의 모든 알람 조건에 적용됩니다. 알람 집합에 대해 시작된 알람 동작은 마지막으로 요청된 동작이 수행되기 전에 조건에 적용될 수 있습니다.

알람 조건이 주기적으로 평가될 경우 특정 알람 동작 수행에 대한 요청은 사용자가 액세스할 수 있는 Progxxx 플래그를 통해 시작된 알람 동작 수행에 대한 요청과 우선 순위가 동일합니다. 이는 알람 동작 수행에 대한 요청이 명령어에 의해 생성된 경우 이 요청은 해당 Progxxx 플래그가 설정된 것처럼 처리되고 ProgXXX 플래그에 지정된 충돌하는 요청을 해결하는 데 사용된 것과 동일한 규칙이 명령어 요청과 Progxxx 플래그를 통해 생성된 요청 사이의 충돌을 해결하는 데 적용된다는 것을 의미합니다.

알람 집합 동작 명령어는 .EnableIn 값이 거짓에서 참으로 전환된 것을 감지한 경우에만 필요한 알람 동작을 시작합니다. 이러한 전환을 감지하기 위해 .LastState 구조 구성원을 사용해 이전 명령어 실행에서 나온 .EnableIn 값을 저장합니다. 위의 실행 섹션을 참조하십시오.

팁: 명령어 파라미터로 제공된 알람 집합에 알람 조건 수가 너무 많을 경우 ASO 명령어의 실행 시간이 크게 증가할 수 있습니다.

추가 참조

알람 명령어 페이지의 29

배열을 통한 인덱스 페이지의 978

비트 명령어

비트 명령어

비트(릴레이 유형) 명령어를 사용하여 입력 비트 또는 타이머 제어 워드 비트 같은 비트의 상태를 모니터링 및 제어합니다.

사용 가능한 명령어

래더 다이어그램

XIC	XIO	OTE	OTL	OTU	ONS	OSR	OSF

펑션 블록과 ST(스트럭처드 텍스트)

OSRI	OSFI

실행할 작업:	사용할 명령어
비트가 설정된 경우 출력 활성화	XIC
비트가 해제된 경우 출력 활성화	XIO
비트 설정	OTE
비트 설정(적산)	OTL
비트 해제(적산)	OTU
렁 참 전환 때마다 1 회 스캔에 대한 출력 활성화	ONS
렁 참 전환 때마다 1 회 스캔에 대한 비트 설정	OSR
렁 거짓 전환 때마다 1 회 스캔에 대한 비트 설정	OSF
펑션 블록에 입력 비트가 설정될 때마다 1 회 스캔에 대한 비트 설정	OSRI
펑션 블록에서 입력 비트가 해제될 때마다 1 회 스캔에 대한 비트 설정	OSFI

추가 참조

비교 명령어 페이지의 329

계산/연산 명령어 페이지의 411

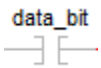
닫힘 검사(XIC)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

XIC 명령어는 데이터 비트를 검사하여 렁 상태를 설정하거나 해제합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

피연산자

래더 다이어그램

피연산자	데이터 유형	형식	설명
Data bit	BOOL	태그	검사할 비트. 데이터 비트에 대해 다양한 피연산자 주소 지정 모드를 사용 가능하며 예를 보려면 *비트 주소 지정* 섹션을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	DataBit 가 참인 경우 렁-출력-조건이 참으로 설정되고, DataBit 가 거짓인 경우렁-출력-조건이 거짓으로 해제됩니다.
사후 스캔	N/A

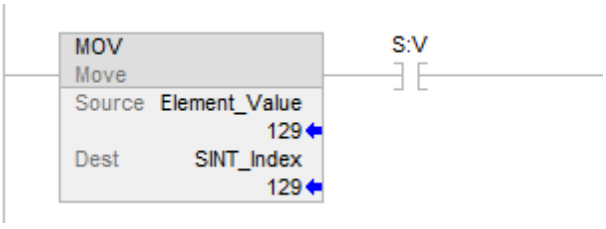
예 1

래더 다이어그램

Limit_Switch_1 이 참인 경우 다음 명령어가 활성화됩니다.

예 2

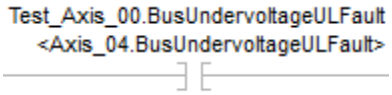
래더 다이어그램



S:V 가 참인 경우(MOV 에 의해 생성됨) 다음 명령어가 활성화됩니다.

예 3

래더 다이어그램



XIC 액세스 LINT 번호

Axis_04 는 AXIS_CIP_DRIVE 태그입니다.

Test_Axis_00 은 Axis_04 의 별칭입니다.

AXIS_CIP_DRIVE 유형에는 CIPAxisFaults 라는 LINT 구성원이 있습니다.

BusUndervoltageULFault 는 CIPAxisFaults 의 비트 구성원입니다.

Test_Axis_00.BusUndervoltageULFault 는 CIPAxisFaults 의 비트 34 입니다. 비트 34 값은 0x400000000 입니다.

Test_Axis_00.BusUndervoltageULFault 가 참인 경우 다음 명령어가 활성화됩니다.

추가 참조

비트 명령어 페이지의 81

비트 주소 지정 페이지의 980

배열을 통한 인덱스 페이지의 978

열림 검사(XIO)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

XIO 명령어는 데이터 비트를 검사하여 렁 상태를 설정하거나 해제합니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

래더 다이어그램

피연산자	데이터 유형	형식	설명
Data bit	BOOL	태그	검사할 비트. 데이터 비트에 대해 다양한 피연산자 주소 지정 모드를 사용 가능하며 예를 보려면 *비트 주소 지정* 섹션을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

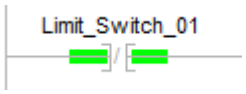
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	데이터 비트가 참인 경우 렁-출력-조건이 거짓으로 해제됩니다. 데이터 비트가 거짓인 경우 렁-출력-조건이 참으로 설정됩니다.
사후 스캔	N/A

예

예 1

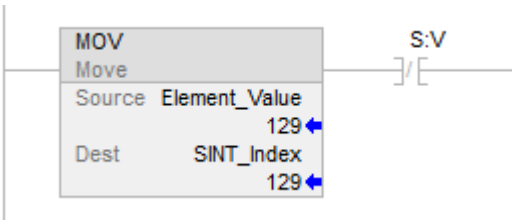
래더 다이어그램



Limit_Switch_01 이 거짓인 경우 다음 명령어가 활성화됩니다.

예 2

래더 다이어그램



S:V 가 거짓인 경우 다음 명령어가 활성화됩니다.

추가 참조

비트 명령어 페이지의 81

비트 주소 지정 페이지의 980

배열을 통한 인덱스 페이지의 978

원샷(ONS)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

렁-입력-조건이 거짓에서 참으로 전환할 때마다 ONS 명령어에 의해 나머지 렁이 참으로 됩니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

피연산자

중요:	다음과 같은 경우 작업 시 예외가 발생할 수 있습니다. • 출력 태그 피연산자가 덮어써집니다. • 구조 피연산자의 구성원이 덮어써집니다. • 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Storage bit	BOOL	태그	내부 저장 비트. 명령어가 마지막으로 실행되었을 때의 렁-입력-조건을 계속 유지합니다. 저장 비트에 대해 다양한 피연산자 주소 지정 모드를 사용 가능하며 사용 예를 보려면 *비트 주소 지정* 섹션을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

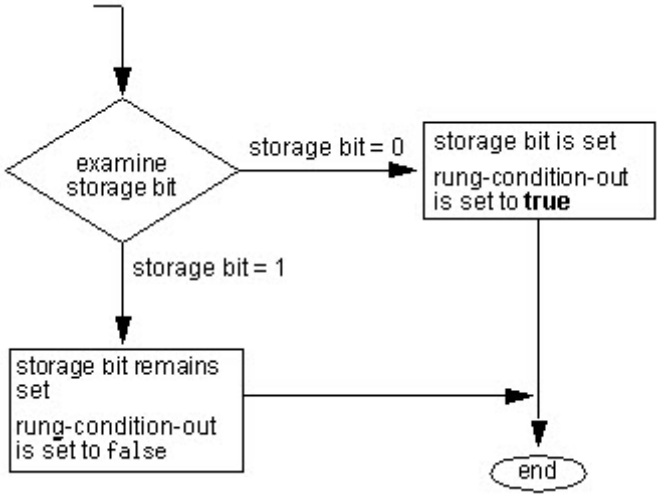
이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

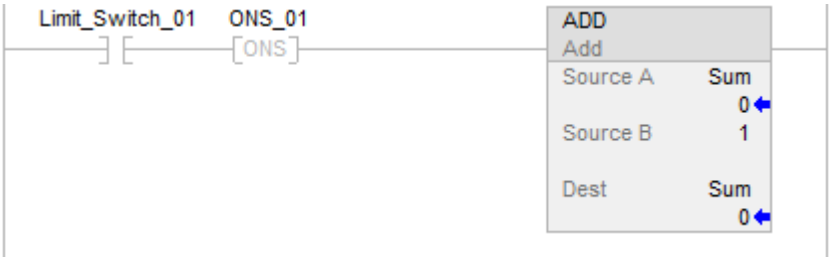
조건/상태	취해진 조치
사전 스캔	첫 번째 스캔 동안 잘못된 트리거가 발생하지 않도록 저장 비트가 참으로 설정됩니다.
렁-입력-조건이 거짓임	저장 비트가 거짓으로 해제되고 렁-출력-조건이 거짓으로 해제됩니다.
렁-입력-조건이 참임	ONS 흐름도(참) 참조하십시오.
사후 스캔	N/A

ONS 흐름도(참)



예

래더 다이어그램



이 예제에서 limit_switch_1 이 거짓에서 참으로 될 때마다 총계가 증가합니다.

추가 참조

비트 명령어 페이지의 81

비트 주소 지정 페이지의 980

배열을 통한 인덱스 페이지의 978

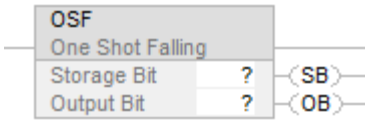
하강 에지 원샷(OSF)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

렁-입력-조건이 참에서 거짓으로 전환할 때 OSF 명령어에 의해 하나의 스캔에 대한 출력 비트가 설정됩니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Storage Bit	BOOL	태그	명령어가 마지막으로 실행되었을 때의 렁-입력-조건이 저장됩니다. 저장 비트에 대해 다양한 피연산자 주소 지정 모드를 사용 가능하며 사용 예를 보려면 비트 주소 지정 섹션을 참조하십시오.
Output Bit	BOOL	태그	수정할 비트.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

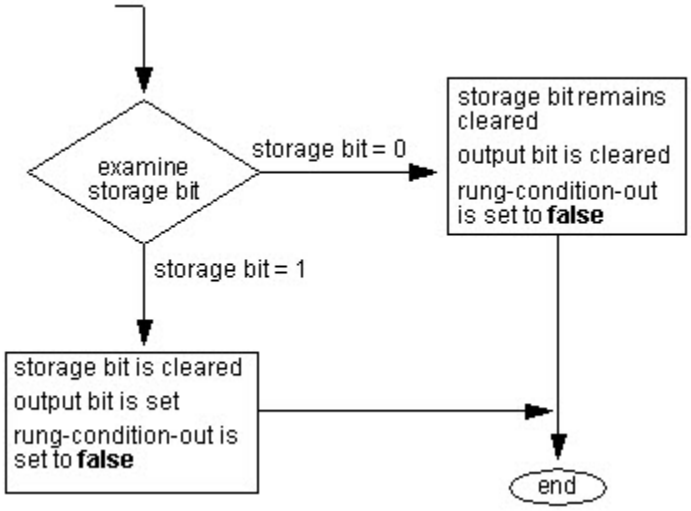
이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

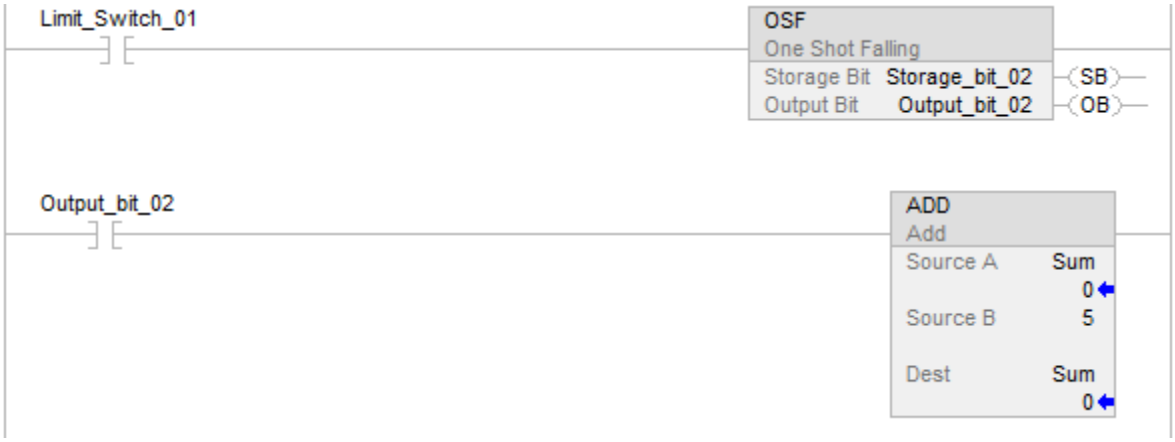
조건/상태	취해진 조치
사전 스캔	첫 번째 프로그램 스캔 동안 잘못된 트리거가 발생하지 않도록 저장 비트가 거짓으로 해제됩니다. 출력 비트가 거짓으로 해제됩니다.
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다. OSF 흐름도(거짓) 참조하십시오.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. 저장 비트가 참으로 설정됩니다. 출력 비트가 거짓으로 해제됩니다.
사후 스캔	N/A

OSF 흐름도(거짓)



예

래더 다이어그램



이 예에서는 OSF 명령어를 사용하여 하나 이상의 명령어를 에지 트리거하는 방법을 보여줍니다. Limit_Switch_01 이 참에서 거짓으로 전환할 때마다 OSF 명령어에 의해 Output_bit_02 가 참으로 설정됩니다. Output_bit_02 를 기준으로 조건화된 명령어가 활성화되고 Output_bit_02 는 하나의 스캔에 대해서만 참이므로 전환 1 회당 명령어가 한 번 실행됩니다.

추가 참조

비트 명령어 페이지의 81

비트 주소 지정 페이지의 980

배열을 통한 인덱스 페이지의 978

입력에서 하강 에지 원샷(OSFI)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

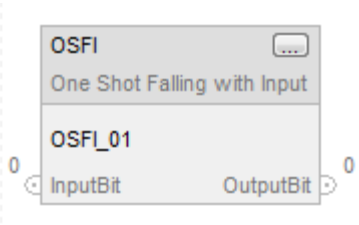
InputBit 가 거짓에서 참으로 전환할 때 OSFI 명령어에 의해 1 회 실행 주기에 대한 OutputBit 가 설정됩니다.

사용 가능한 언어

래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

평션 블록



ST(스트럭처드 텍스트)

OSFI(OSFI_tag)

피연산자

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
OSFI tag	FBD_ONESHOT	구조	OSFI 구조

피연산자 관련 폴트를 보려면 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

평션 블록

피연산자	유형	형식	설명
OSFI tag	FBD_ONESHOT	구조	OSFI 구조

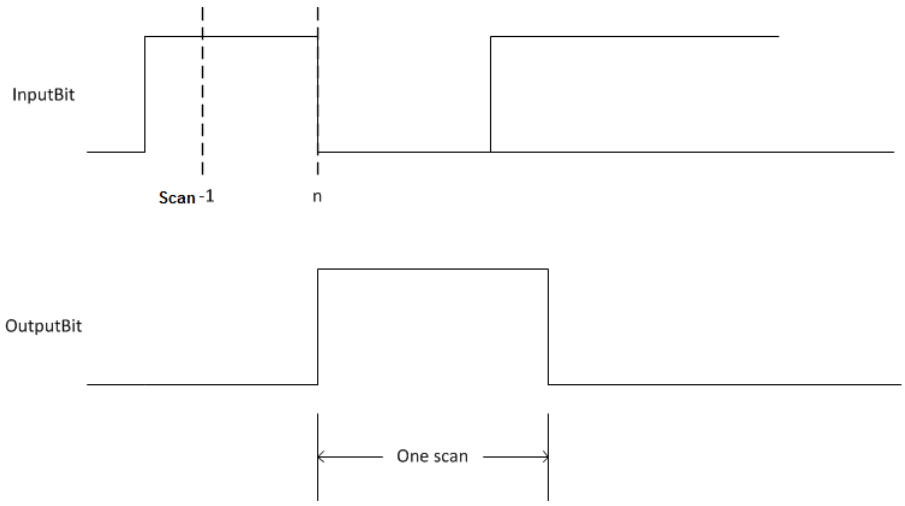
FBD_ONESHOT 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
InputBit	BOOL	입력 비트.

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
OutputBit	BOOL	출력 비트

설명

InputBit 가 거짓이고 명령어가 마지막으로 스캔되었을 때 참이었으면 OutputBit 가 설정되고 그렇지 않으면 OutputBit 가 해제됩니다.



연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

펑션 블록

조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 명령어가 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	InputBit 의 참~거짓 전환을 요청하기 위해 이전 InputBit 기록을 지웁니다.
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	펑션 블록 표의 사전 스캔을 참조하십시오.
정상 실행	펑션 블록 표에서 Tag.EnableIn 이 참임을 참조하십시오.
사후 스캔	펑션 블록 표에서 사후 스캔을 참조하십시오.

예

limit_switch1 이 설정에서 해제로 되면 OSFI 명령어에 의해 하나의 스캔에 대한 OutputBit 가 설정됩니다.

평션 블록

ST(스트럭쳐드 텍스트)

```
OSFI_01.InputBit := limit_switch1;

OSFI(OSFI_01);

Output_state := OSFI_01.OutputBit;
```

추가 참조

비트 명령어 페이지의 81

OSF 페이지의 89

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

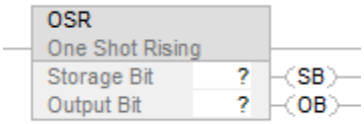
상승 에지 원샷(OSR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

렁-입력-조건이 거짓에서 참으로 전환할 때 OSR 명령어에 의해 하나의 스캔에 대한 출력 비트가 설정됩니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

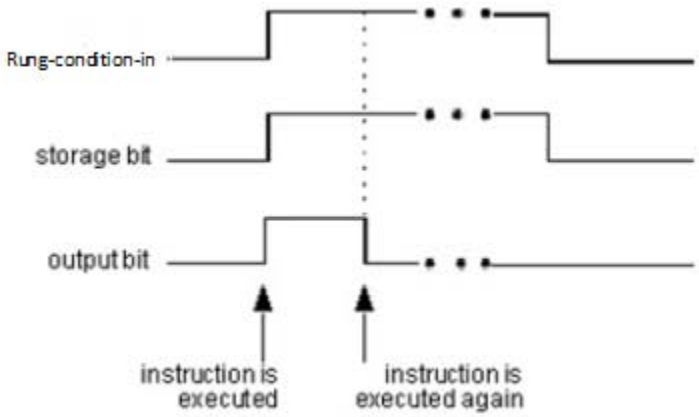
중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Storage Bit	BOOL	태그	명령어가 마지막으로 실행되었을 때의 렁-입력-조건이 저장됩니다. 저장 비트에 대해 다양한 피연산자 주소 지정 모드를 사용 가능하며 사용 예를 보려면 *비트 주소 지정* 섹션을 참조하십시오.
Output Bit	BOOL	태그	수정할 비트.

설명



연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

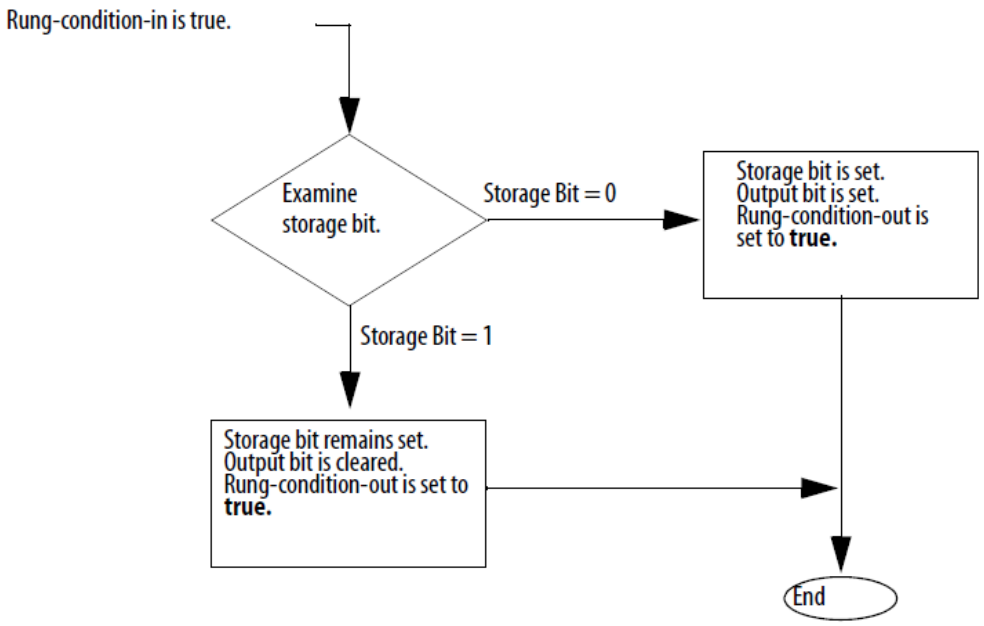
이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	첫 번째 프로그램 스캔 동안 잘못된 트리거가 발생하지 않도록 저장 비트를 참으로 설정합니다. 출력 비트가 거짓으로 해제됩니다.
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정 저장 비트가 거짓으로 해제됩니다. 출력 비트가 거짓으로 해제됩니다.
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 OSR 흐름도(참) 참조하십시오.
사후 스캔	N/A

OSR 흐름도(참)



예

래더 다이어그램

이 예에서는 OSR 명령어를 사용하여 하나 이상의 명령어를 에지 트리거하는 방법을 보여줍니다. Limit_Switch_01 이 거짓에서 참으로 전환할 때마다 OSR 명령어에 의해 Output_bit_02 가 참으로 설정됩니다. Output_bit_02 를 기준으로 조건화된 명령어가 활성화되고 Output_bit_02 는 하나의 스캔에 대해서만 참이므로 전환 1 회당 명령어가 한 번 실행됩니다.

추가 참조

비트 명령어 페이지의 81

비트 주소 지정 페이지의 980

배열을 통한 인덱스 페이지의 978

입력에서 상승 에지 원샷(OSRI)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

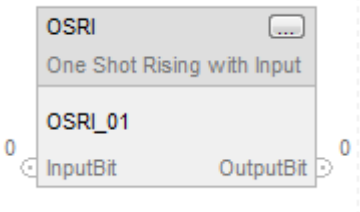
입력 비트가 해제에서 설정으로 전환할 때 OSRI 명령어에 의해 1 회 실행 주기에 대한 출력 비트가 설정됩니다.

사용 가능한 언어

래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

평션 블록



ST(스트럭처드 텍스트)

OSRI(OSRI_tag);

피연산자

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
OSRI tag	FBD_ONESHOT	구조	OSRI 구조

평션 블록

피연산자	유형	형식	설명
OSRI tag	FBD_ONESHOT	구조	OSRI 구조

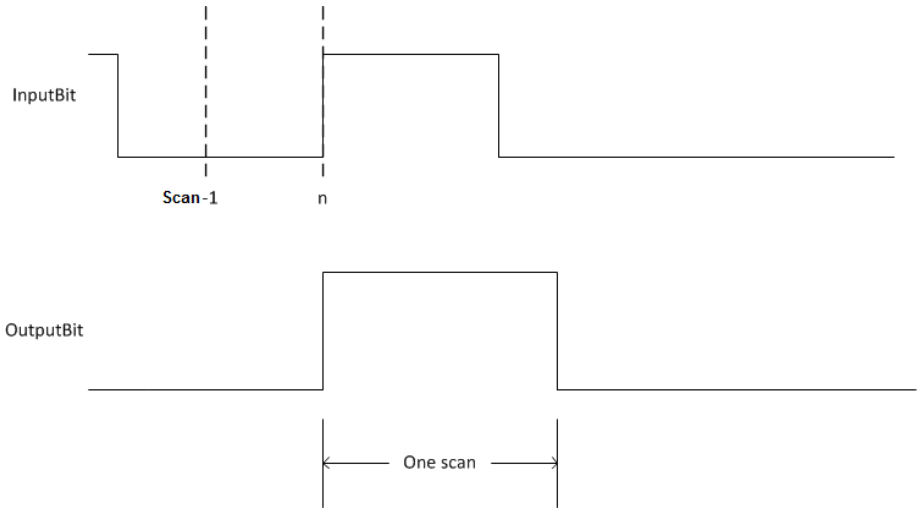
FBD_ONESHOT 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	선택하지 않으면 명령이 실행되지 않고 출력이 업데이트되지 않습니다. 설정되면 명령어가 실행됩니다. 기본값은 설정 상태입니다.
InputBit	BOOL	입력 비트. 기본값은 해제 상태입니다.

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
OutputBit	BOOL	출력 비트

설명

InputBit 가 참이고 마지막으로 명령어가 스캔되었을 때 거짓이었으면 OutputBit 가 설정되고 그렇지 않으면 OutputBit 가 해제됩니다.



연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

펑션 블록

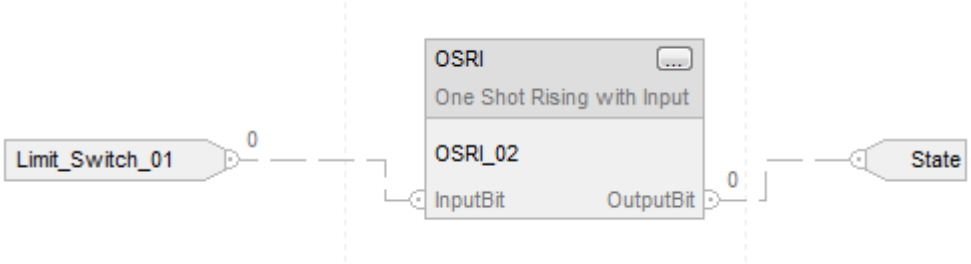
조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.Enable-in 이 거짓	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.Enable-in 이 참	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 명령어가 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	InputBit 의 거짓-참 전환을 요청하려면 이전 InputBit 기록이 설정됩니다.
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	펑션 블록 표의 사전 스캔을 참조하십시오.
정상 실행	펑션 블록 표에서 Tag.EnableIn 이 참임을 참조하십시오.
사후 스캔	펑션 블록 표의 사후 스캔 참조하십시오.

예

평션 블록



limit_switch1 이 해제 상태에서 설정으로 되면 OSRI 명령어에 의해 하나의 스캔에 대한 OutputBit 가 설정됩니다.

ST(스트럭쳐드 텍스트)

```
OSRI_01.InputBit := limit_switch1;

OSRI(OSRI_01);

State := OSRI_O1.OutputBit;
```

추가 참조

비트 명령어 페이지의 81

하강 에지 원샷(OSF) 페이지의 89

원샷(ONS) 페이지의 87

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

출력 전원 공급(OTE)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

OTE 명령어는 렁 상태를 기준으로 데이터 비트를 설정 또는 해제합니다.

사용 가능한 언어

래더 다이어그램

펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Data bit	BOOL	태그	수정할 비트. 데이터 비트에 대해 다양한 피연산자 주소 지정 모드를 사용 가능하며 예를 보려면 *비트 주소 지정* 섹션을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	데이터 비트가 거짓으로 해제됩니다.
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다. 데이터 비트가 거짓으로 해제됩니다.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. 데이터 비트가 참으로 설정됩니다.
사후 스캔	데이터 비트가 거짓으로 해제됩니다.

예

래더 다이어그램

스위치가 참인 경우 OTE 명령어에 의해 Light_01 이 참으로 설정됩니다. 스위치가 거짓인 경우 OTE 명령어에 의해 Light_01 이 거짓으로 해제됩니다.

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

비트 명령어 페이지의 81

비트 주소 지정 페이지의 980

배열을 통한 인덱스 페이지의 978

출력 래치(OTL)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

OTL 명령어는 데이터 비트를 설정(래치)합니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Data bit	BOOL	태그	수정할 비트. 데이터 비트에 대해 다양한 피연산자 주소 지정 모드를 사용 가능하며 예를 보려면 *비트 주소 지정* 섹션을 참조하십시오.

설명

렁 상태가 참인 경우 OTL 명령어에 의해 데이터 비트가 참으로 설정됩니다. 일반적으로 OTU 명령어에 의해 해제될 때까지 데이터 비트는 계속 참입니다. 렁 상태가 거짓으로 변경되어도 OTL 명령어로 인해 데이터 비트의 상태가 변경되지 않습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580 및 GuardLogix 5580 컨트롤러의 경우 피연산자가 간접 배열 참조이고 첨자가 범위를 벗어나는 경우 OTL 명령어가 거짓일 때 컨트롤러에서 메이저 폴트가 생성되지 않습니다.

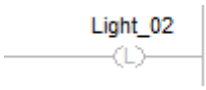
실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. 데이터 비트가 참으로 설정됩니다.
사후 스캔	N/A

예

래더 다이어그램



활성화된 경우 OTL 명령어에 의해 라이트가 켜집니다.

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

비트 명령어 페이지의 81

비트 주소 지정 페이지의 980

배열을 통한 인덱스 페이지의 978

출력 래치 해제(OTU)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

OTU 명령어는 데이터 비트를 해제(래치 해제)합니다.

사용 가능한 언어

래더 다이어그램

펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요:	다음과 같은 경우 작업 시 예외가 발생할 수 있습니다. • 출력 태그 피연산자가 덮어써집니다. • 구조 피연산자의 구성원이 덮어써집니다. • 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Data bit	BOOL	태그	수정할 비트. 데이터 비트에 대해 다양한 피연산자 주소 지정 모드를 사용 가능하며 예를 보려면 *비트 주소 지정* 섹션을 참조하십시오.

설명

렁 상태가 참인 경우 OTU 명령어에 의해 데이터 비트가 거짓으로 해제됩니다. 렁 상태가 거짓으로 변경되어도 OTU 명령어에 의해 데이터 비트 상태가 변경되지 않습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 데이터 비트가 거짓으로 해제됩니다.
사후 스캔	N/A

예

래더 다이어그램

활성화되면 OTU 명령어에 의해 Light_02 가 해제됩니다.

추가 참조

비트 명령어 페이지의 81

비트 주소 지정 페이지의 980

배열을 통한 인덱스 페이지의 978

타이머 및 카운터 명령어

타이머 및 카운터 명령어

타이머와 카운터는 시간 또는 이벤트의 수를 기준으로 작업을 제어합니다.

사용 가능한 명령어

래더 다이어그램

TON	TOF	RTO	CTU	CTD	RES

펑션 블록과 ST(스트럭처드 텍스트)

TONR	TOFR	RTOR	CTUD

실행할 작업	사용할 명령어
타이머가 활성화된 시간 측정	TON
타이머가 비활성화된 시간 측정	TOF
시간 누산	RTO
펑션 블록에 리셋이 내장된 상태에서 타이머가 활성화된 시간 측정	TONR
펑션 블록에 리셋이 내장된 상태에서 타이머가 비활성화된 시간 측정	TOFR
펑션 블록에 리셋이 내장된 상태에서 누산 시간 측정	RTOR
위로 카운트	CTU
아래로 카운트	CTD
펑션 블록의 위로 카운트 및 아래로 카운트	CTUD
타이머 또는 카운터 리셋	RES

모든 타이머에 대한 시간 기준은 1 밀리초입니다. 예를 들어 2 초 타이머의 .PRE 값은 2000 입니다.

추가 참조

계산/연산 명령어 페이지의 411

비교 명령어 페이지의 329

비트 명령어 페이지의 81

ASCII 문자열 명령어 페이지의 907

ASCII 변환 명령어 페이지의 927

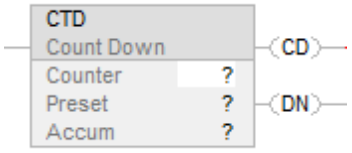
아래로 카운트(CTD)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

CTD 명령어는 렁-입력-조건이 거짓에서 참으로 전환할 때마다 카운트다운합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Counter	COUNTER	태그	카운터 구조
Preset	DINT	즉시	Counter.PRE 값.
Accum	DINT	즉시	Counter.ACC 값.

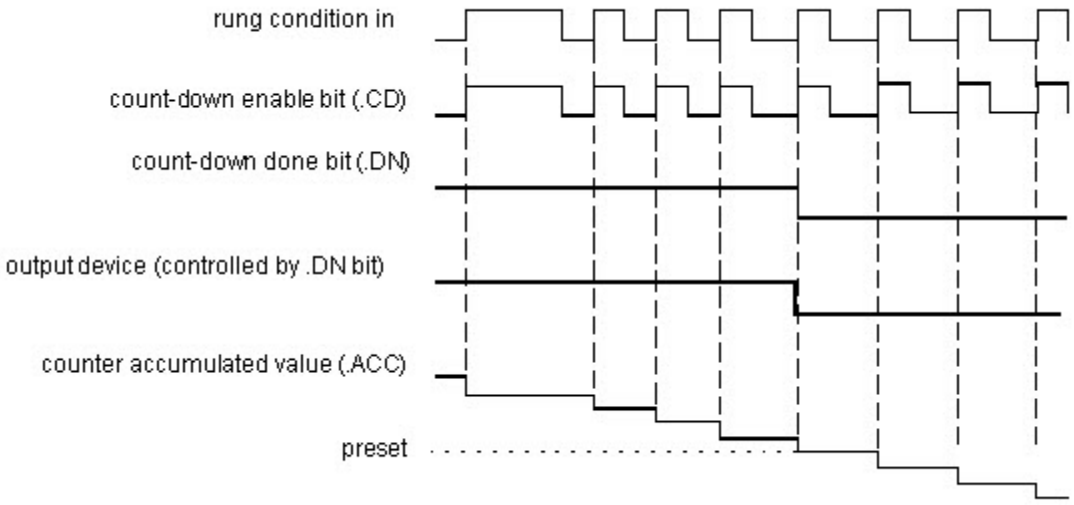
COUNTER 구조

니모닉	데이터 유형	설명
.CD	BOOL	카운트다운 활성화 비트에는 명령어가 마지막으로 실행되었을 때 렁-입력-조건이 들어 있습니다.
.DN	BOOL	완료 비트로 해제되면 카운팅 작업이 완료되었음을 나타냅니다.
.OV	BOOL	오버플로 비트로 설정된 경우 카운터가 상한인 2,147,483,647 을 지나 증가했음을 나타냅니다.
.UN	BOOL	언더플로 비트로 설정된 경우 카운터가 하한인 -2,147,483,648 을 지나 감소했음을 나타냅니다.
.PRE	DINT	사전 설정 값으로 명령어가 완료되었음을 나타내기 위해 누산 값이 도달해야 하는 값을 지정합니다.
.ACC	DINT	누산 값으로 명령어에 의해 카운트된 전환의 수를 지정합니다.

설명

CTD 명령어는 일반적으로 동일한 카운터 구조를 참조하는 CTU 명령어와 함께 사용됩니다.

렁-입력-조건이 참으로 설정되고 .CD 가 거짓일 경우 .ACC 는 1 씩 감소합니다. 렁-입력-조건이 거짓일 경우 .CD 는 거짓으로 해제됩니다.



연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

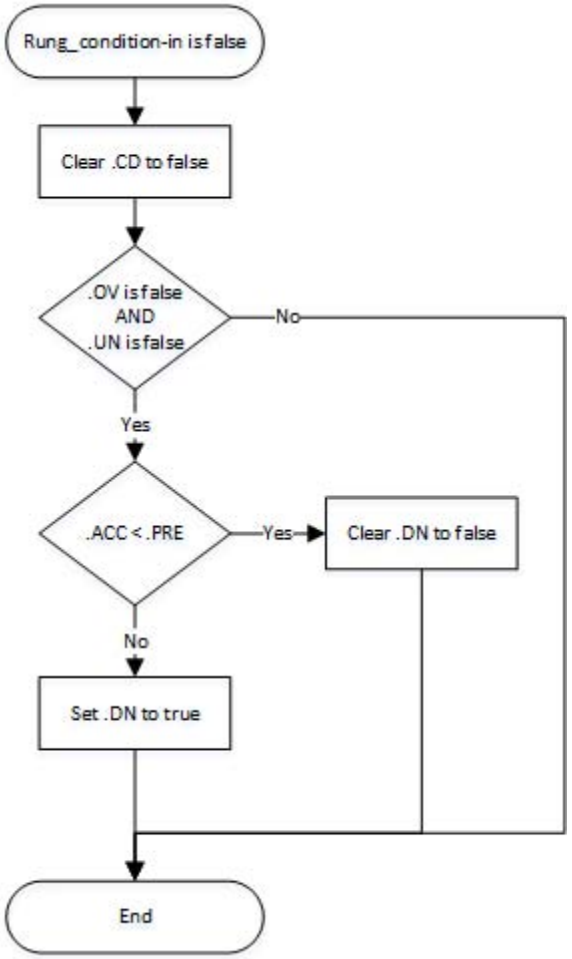
이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

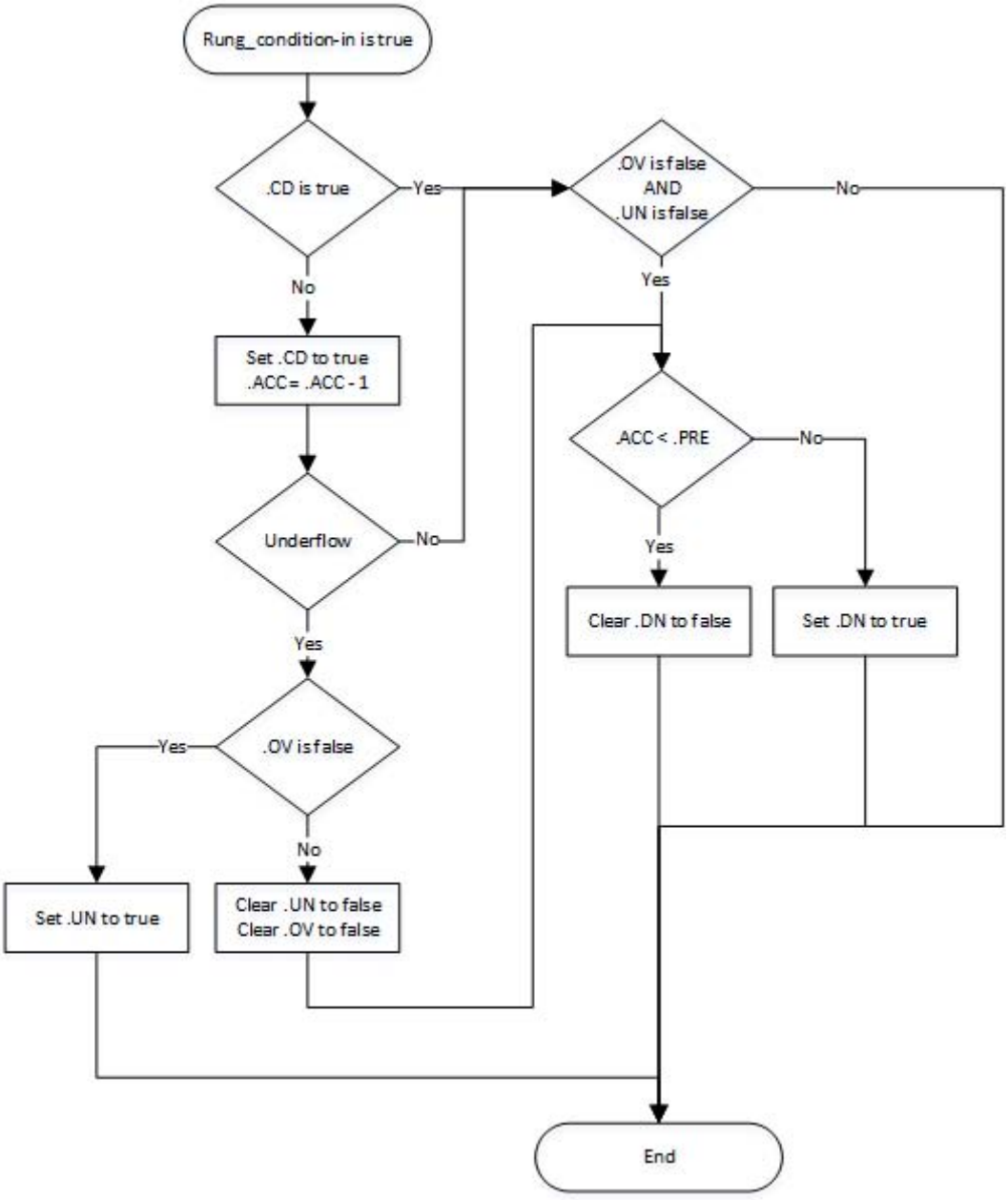
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	.CD 비트는 첫 번째 프로그램 스캔 동안 유효하지 않은 감분이 일어나지 않도록 참으로 설정됩니다.
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정 CTD 흐름도(거짓) 참조하십시오.
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 CTD 흐름도(참) 참조하십시오.
사후 스캔	N/A

CTD 흐름도(거짓)

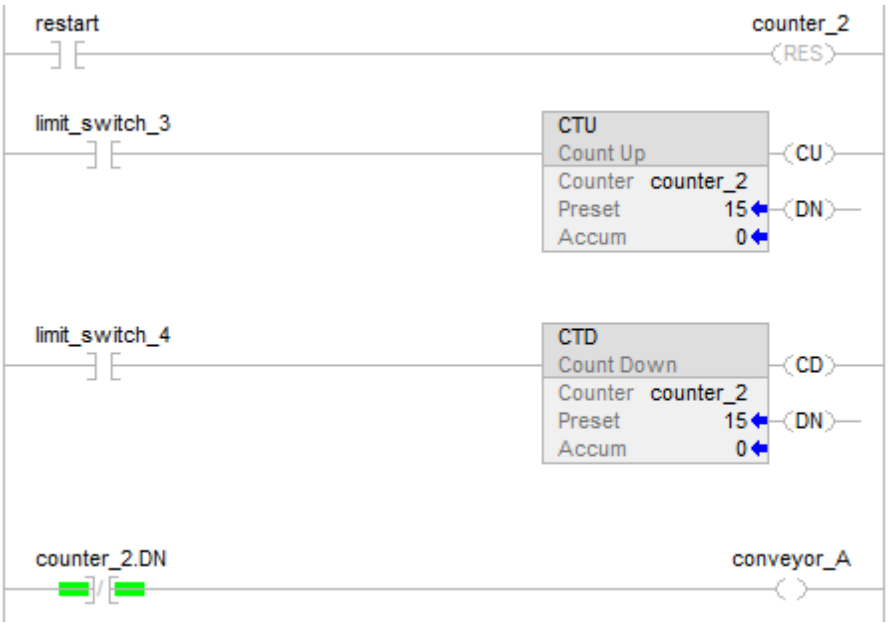


CTD 흐름도(참)


Rung_condition-in is true
.CD is true
Yes
No
.OV is false AND .UN is false
No
Yes
Set .CD to true
.ACC = .ACC - 1
.ACC < .PRE
No
Yes
Underflow
No
Yes
Clear .DN to false
Set .DN to true
Yes
.OV is false
Yes
No
Set .UN to true
Clear .UN to false
Clear .OV to false
End

예

래더 다이어그램



컨베이어가 부품을 버퍼 구역으로 이동시킵니다. 부품이 반입될 때마다 limit_switch_3 이 활성화되고 counter_2 가 1 씩 증가합니다. 부품이 나갈 때마다 limit_switch_4 가 활성화되고 counter_2 가 1 씩 감소합니다. 버퍼 구역에 100 개의 부품이 있는 경우(counter_2.dn 이 참) conveyor_A 가 켜짐으로 전환되고 버퍼에 추가로 부품을 반입할 여유 공간이 생길 때까지 컨베이어의 부품 추가 반입이 중단됩니다.

추가 참조

배열을 통한 인덱스 페이지의 978

카운터 명령어 페이지의 111

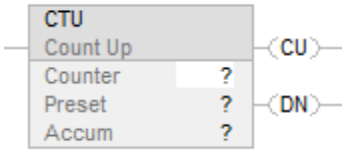
위로 카운트(CTU)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

렁-입력-조건이 거짓에서 참으로 전환할 때마다 CTU 명령어에 의해 카운트업됩니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

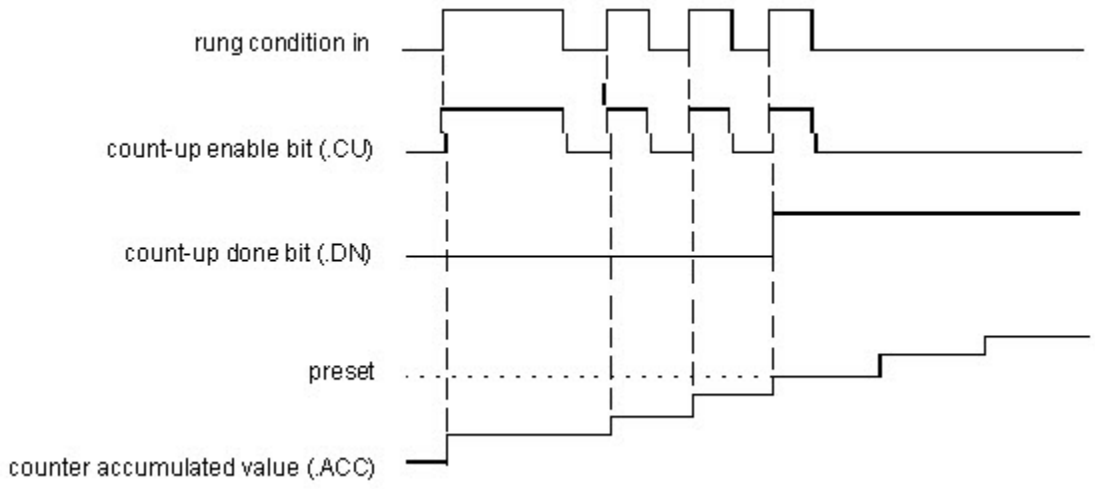
피연산자	데이터 유형	형식	설명
Counter	COUNTER	태그	카운터 구조
Preset	DINT	즉시	Counter.PRE 값.
Accum	DINT	즉시	Counter.ACC 값.

COUNTER 구조

니모닉	데이터 유형	설명
.CU	BOOL	위로 카운트 활성화 비트에는 명령어가 마지막으로 실행되었을 때 렁-입력-조건이 들어 있습니다.
.DN	BOOL	완료 비트는 설정된 경우 카운팅 작업이 완료되었음을 나타냅니다.
.OV	BOOL	오버플로 비트로 설정된 경우 카운터가 상한인 2,147,483,647 을 지나 증가했음을 나타냅니다.
.UN	BOOL	언더플로 비트로 설정된 경우 카운터가 하한인 -2,147,483,648 을 지나 감소했음을 나타냅니다.
.PRE	DINT	사전 설정 값으로 명령어가 완료되었음을 나타내기 위해 누산 값이 도달해야 하는 값을 지정합니다.
.ACC	DINT	누산 값으로 명령어에 의해 카운트된 전환의 수를 지정합니다.

설명

렁-입력-조건이 참으로 설정되고 .CU 가 거짓일 경우 .ACC 는 1 씩 증가합니다. 렁-입력-조건이 거짓일 경우 .CU 는 거짓으로 해제됩니다.



연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

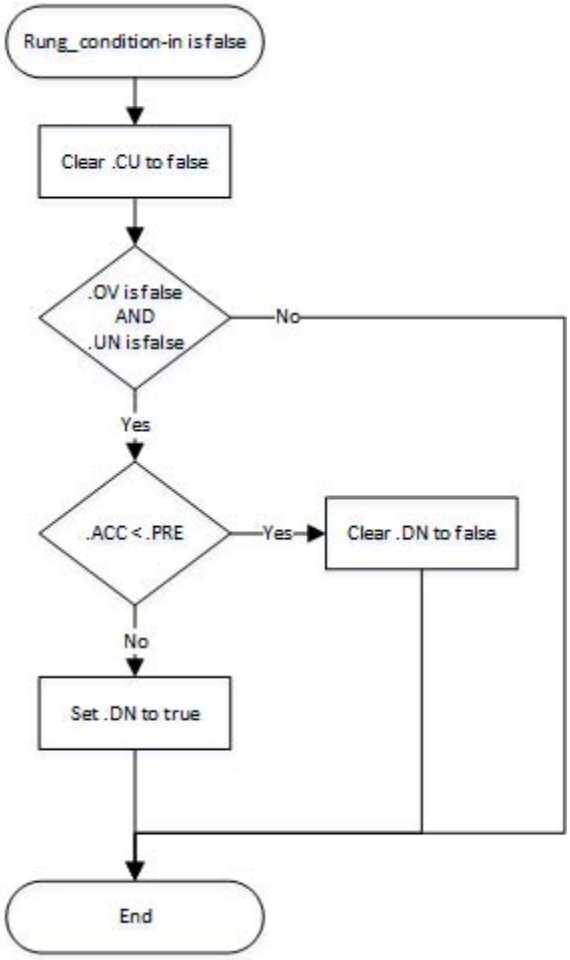
이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

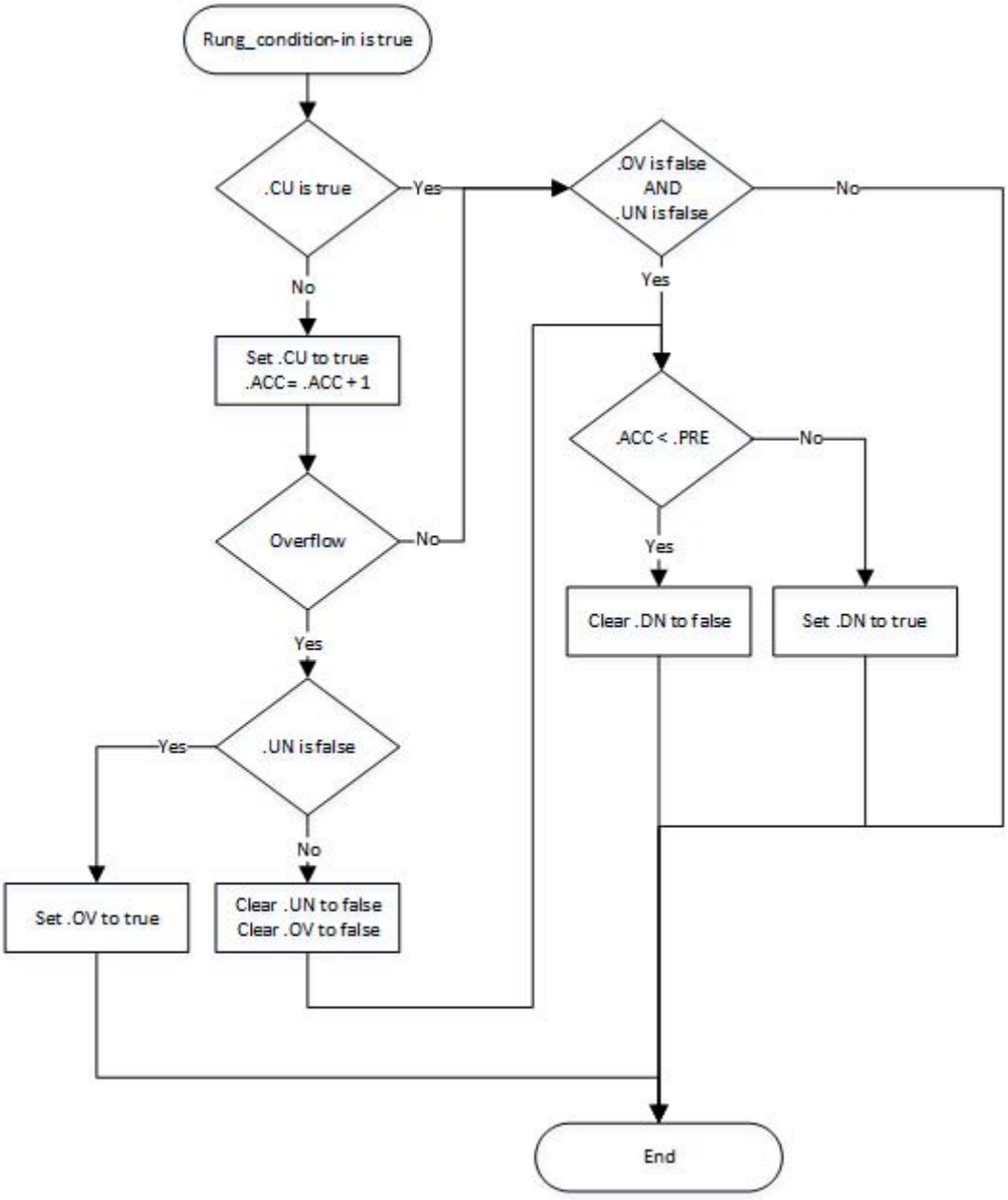
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	.CU 비트는 첫 번째 프로그램 스캔 동안 유효하지 않은 증가가 발생하지 않도록 참으로 설정됩니다.
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정 CTU 흐름도(거짓) 참조하십시오.
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 CTU 흐름도(참) 참조하십시오.
사후 스캔	N/A

CTU 흐름도(거짓)

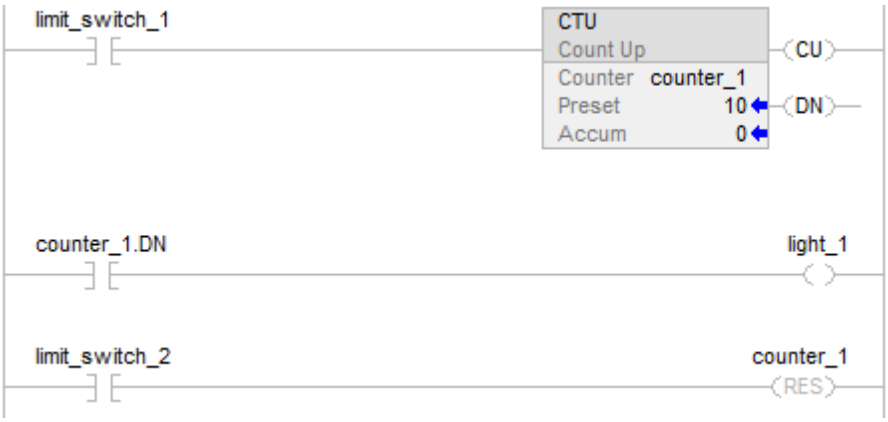

Rung_condition-in is false
Clear .CU to false
.OV is false
AND
.UN is false
No
Yes
.ACC < .PRE
Yes
Clear .DN to false
No
Set .DN to true
End

CTU 흐름도(참)


Rung_condition-in is true
.CU is true
Yes
No
.OV is false
AND
.UN is false
No
Yes
Set .CU to true
.ACC = .ACC + 1
.ACC < .PRE
No
Yes
Overflow
No
Yes
Clear .DN to false
Set .DN to true
Yes
.UN is false
Yes
No
Set .OV to true
Clear .UN to false
Clear .OV to false
End

예

래더 다이어그램



limit_switch_1 이 비활성 상태에서 10 회 활성화된 후 .DN 비트가 참으로 설정되고 light_1 이 켜짐으로 전환됩니다. limit_switch_1 이 계속 비활성 상태에서 활성화되는 경우 counter_1 은 계속 증가하고 .DN 비트는 계속 설정되어 있습니다. limit_switch_2 가 활성화된 경우 RES 명령어에 의해 counter_1 이 리셋되고(상태 비트 및 .ACC 값 해제) light_1 이 꺼짐으로 전환됩니다.

추가 참조

배열을 통한 인덱스 페이지의 978

카운터 명령어 페이지의 111

위로/아래로 카운트(CTUD)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

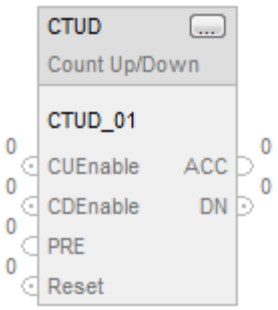
CTUD 명령어는 CUEnable 이 해제에서 설정으로 전환할 때 1 씩 카운트업합니다. CDEnable 이 해제에서 설정으로 전환할 때는 명령어에 의해 1 씩 카운트다운됩니다.

사용 가능한 언어

래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

평션 블록



ST(스트럭처드 텍스트)

CTUD(CTUD_tag)

피연산자

ST(스트럭처드 텍스트)

변수	유형	형식	설명
CTUD tag	FBD_COUNTER	구조	CTUD 구조

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 ST(스트럭처드 텍스트) 구문을 참조하십시오.

평션 블록

피연산자	유형	형식	설명
CTUD tag	FBD_COUNTER	구조	CTUD 구조

FBD_COUNTER 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	선택하지 않으면 명령이 실행되지 않고 출력이 업데이트되지 않습니다. 설정되면 명령어가 실행됩니다. 기본값은 설정 상태입니다.

CUEnable	BOOL	카운트업 활성화. 입력이 해제에서 설정으로 전환할 때 누적기가 1 씩 카운트업됩니다. 기본값은 해제 상태입니다.
CDEnable	BOOL	카운트다운 활성화. 입력이 해제에서 설정으로 전환할 때 누적기가 1 씩 카운트다운됩니다. 기본값은 해제 상태입니다.
PRE	DINT	카운터 사전 설정 값. DN 으로 설정되기 전 누산 값이 도달해야 하는 값입니다. 유효값 = 모든 정수 기본값은 0 입니다.
Reset	BOOL	타이머 리셋 요청. 설정된 경우 카운터가 리셋됩니다. 기본값은 해제 상태입니다.

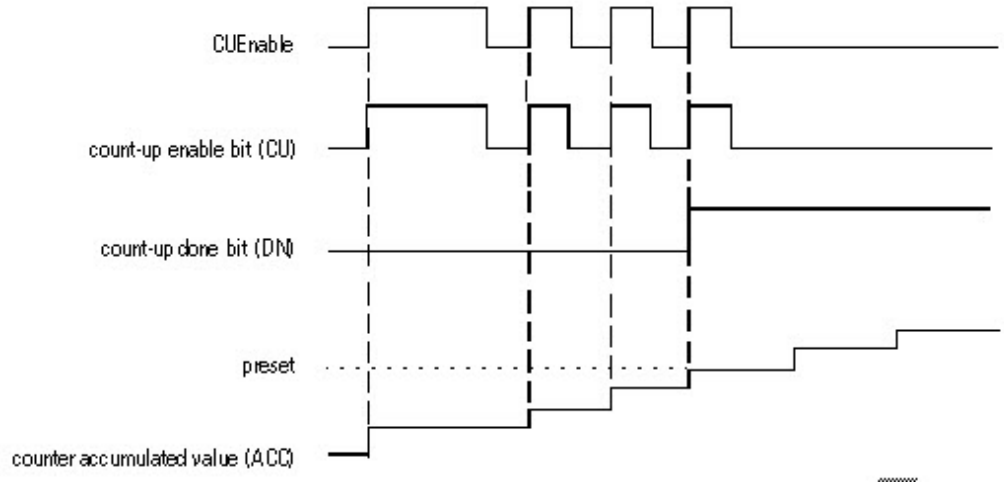
출력 파라미터	**데이터 유형**	**설명**
EnableOut	BOOL	명령어에 의해 유효한 결과가 산출됩니다.
ACC	DINT	누산 값.
CU	BOOL	위로 카운트이 활성화됩니다.
CD	BOOL	아래로 카운트이 활성화됩니다.
DN	BOOL	카운팅 완료. 누산 값이 사전 설정값보다 크거나 같을 경우 설정됩니다.
OV	BOOL	카운터 오버플로. 카운터가 상한인 2,147,483,647 을 초과했음을 나타냅니다. 그러면 카운터가 -2,147,483,648 로 롤오버되고 카운트다운을 다시 시작합니다.
UN	BOOL	카운터 언더플로. 카운터가 하한인 -2,147,483,648 을 초과했음을 나타냅니다. 그러면 카운터가 2,147,483,647 로 되돌아가서 다시 카운트다운을 시작합니다.

설명

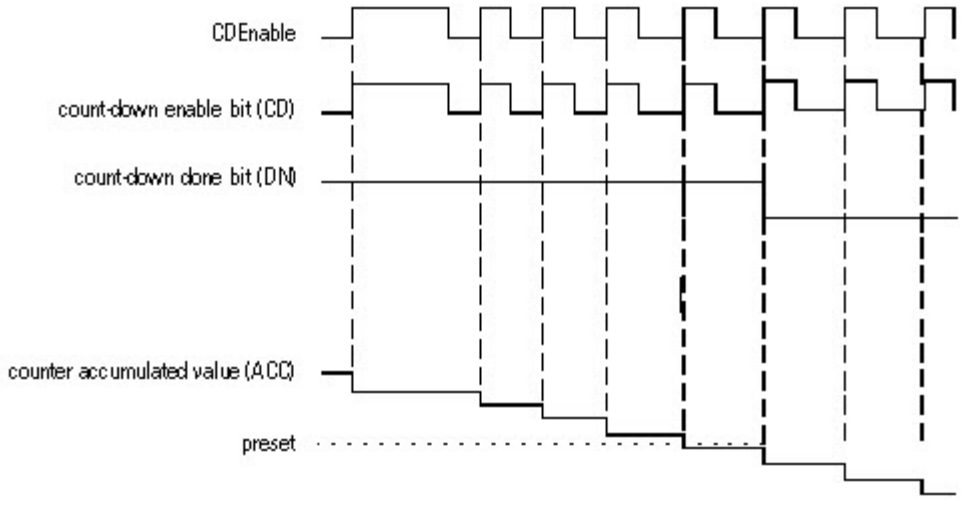
참이고 CUEnable 이 참일 때 CTUD 명령어에 의해 카운터가 1 씩 증가합니다. 참이고 CDEnable 이 참일 때 CTUD 명령어에 의해 카운터가 1 씩 감소합니다.

CUEnable 및 CDEnable 입력 파라미터 모두 동일한 스캔 도중에 전환 가능합니다. 명령어는 아래로 카운트에 앞서 위로 카운트을 실행합니다.

위로 카운트



아래로 카운트



비활성화된 경우 CTUD 명령어에 의해 누산 값이 그대로 유지됩니다. 명령어를 리셋하려면 Reset 입력 파라미터를 설정합니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

평션 블록

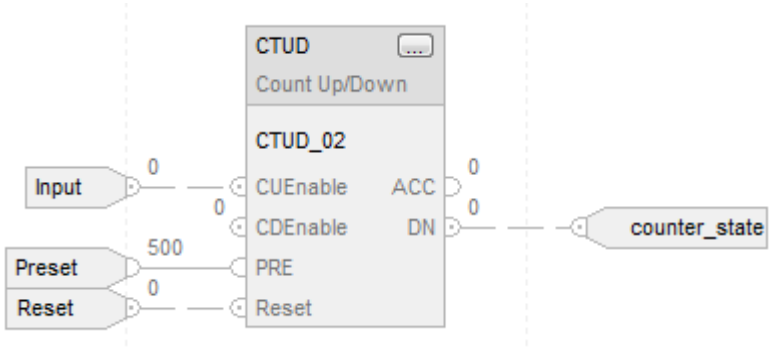
조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다. CuEnable 또는 CdEnable 파라미터의 "0 ~ 1" 전환을 요청하여 ACC 를 적용하려면 데이터를 초기화합니다.
Tag.EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 명령어가 실행됩니다.
명령어 최초 실행	CuEnable 또는 CdEnable 파라미터의 "0 ~ 1" 전환을 요청하여 ACC 를 적용하려면 데이터를 초기화합니다.
명령어 최초 스캔	CuEnable 또는 CdEnable 파라미터의 "0 ~ 1" 전환을 요청하여 ACC 를 적용하려면 데이터를 초기화합니다.
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	펑션 블록 표의 사전 스캔을 참조하십시오.
정상 실행	펑션 블록 표에서 Tag.EnableIn 이 참임을 참조하십시오.
사후 스캔	펑션 블록 표에서 사후 스캔을 참조하십시오.

예

펑션 블록



ST(스트럭처드 텍스트)

```
CTUD_01.PRE := 500;

CTUD_01.Reset := Reset;

CTUD_01.CUEnable := Input;

CTUD(CTUD_01);

counter_state := CTUD_01.DN;
```

추가 참조

공통 속성 페이지의 963

위로 카운트(CTU) 페이지의 117

아래로 카운트(CTD) 페이지의 112

리셋(RES) 페이지의 129

ST(스트럭쳐드 텍스트) 구문 페이지의 997

리셋(RES)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

RES 명령어에 의해 TIMER, COUNTER 또는 CONTROL 구조가 리셋됩니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Structure	TIMER CONTROL COUNTER	태그	리셋 대상 구조

설명

참일 경우 RES 명령어에 의해 다음 요소가 해제됩니다.

RES 명령어 사용 대상	명령어에 의해 해제되는 항목
TIMER	.ACC 값이 0 으로 설정됨 제어 상태 비트가 거짓으로 해제됨
COUNTER	.ACC 값이 0 으로 설정됨 제어 상태 비트가 거짓으로 해제됨
CONTROL	.POS 값이 0 으로 설정됨 제어 상태 비트가 거짓으로 해제됨

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

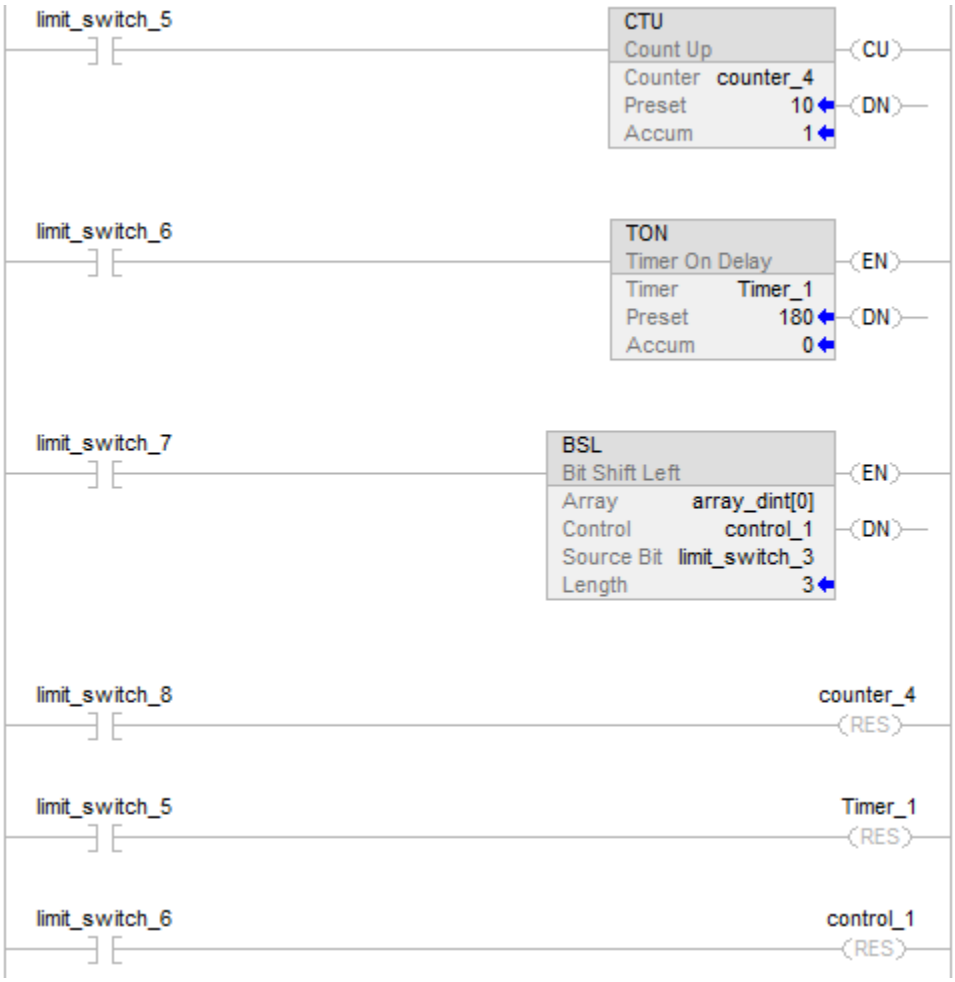
실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. 지정된 구조가 리셋됩니다.
사후 스캔	N/A

예

래더 다이어그램



리셋 예

상기 예에서:

limit_switch_8 이 활성화될 때 counter_4 리셋

limit_switch_5 가 활성화될 때 Timer_1 리셋

limit_switch_6 이 활성화될 때 control_1 리셋

추가 참조

카운터 명령어 페이지의 111

배열을 통한 인덱스 페이지의 978

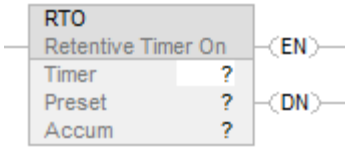
켜짐 적산 타이머(RTO)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

RTO 명령어는 명령어가 활성화된 시간을 누산하는 적산 타이머입니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Timer	TIMER	태그	타이머 구조
Preset	DINT	즉시	Timer.PRE 값.
Accum	DINT	즉시	Timer.ACC 값.

TIMER 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트에는 명령어가 마지막으로 실행되었을 때 렁-입력-조건이 들어 있습니다.
.TT	BOOL	타이밍 비트는 설정된 경우 타이밍 작업이 진행 중임을 나타냅니다.
.DN	BOOL	완료 비트는 설정된 경우 타이밍 작업이 완료(또는 일시 중지)되었음을 나타냅니다.
.PRE	DINT	사전 설정 값으로 명령어가 완료된 것으로 나타나기 전에 누산 값이 도달해야 하는 값(1 밀리초 단위)을 지정합니다.
.ACC	DINT	누산 값은 RTO 명령어가 활성화된 이후로 경과한 시간을 밀리초로 지정합니다.

설명

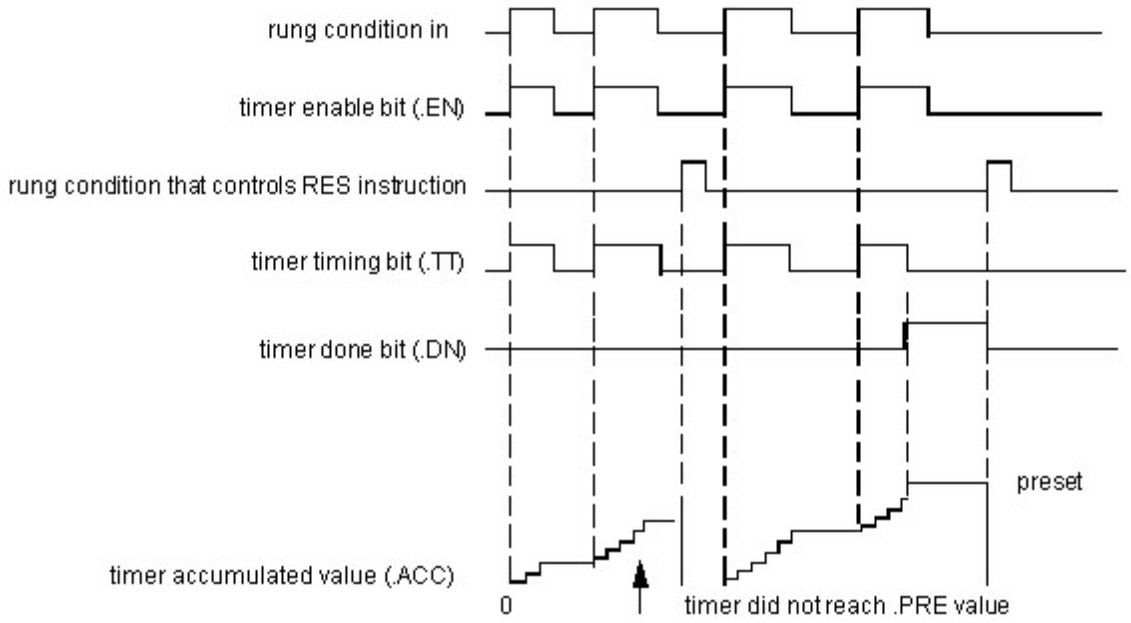
RTO 명령어는 다음 시점에 이르는 시간을 누산합니다.

- 타이머가 비활성화됨
- 타이머 완료

시간 기준은 항상 1 밀리초입니다. 예를 들어 2 초 타이머의 경우 .PRE 값으로 2000 을 입력합니다.

타이머는 타이머가 완료할 때 .DN 비트를 참으로 설정합니다.

활성화되었을 때 .DN 비트를 참으로 설정하여 타이밍을 일시 중단한 후에 다시 거짓으로 해제하여 타이밍을 다시 시작할 수 있습니다.



타이머 실행 방식

타이머는 현재 시간에서 타이머의 마지막 스캔 시간을 빼는 식으로 실행됩니다.

ACC = ACC + (current_time - last_time_scanned)

ACC 를 업데이트한 후에는 타이머가 last_time_scanned = current_time 으로 설정되고, 다음 스캔에 대비하여 준비됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
.PRE < 0	4	34
.ACC < 0	4	34

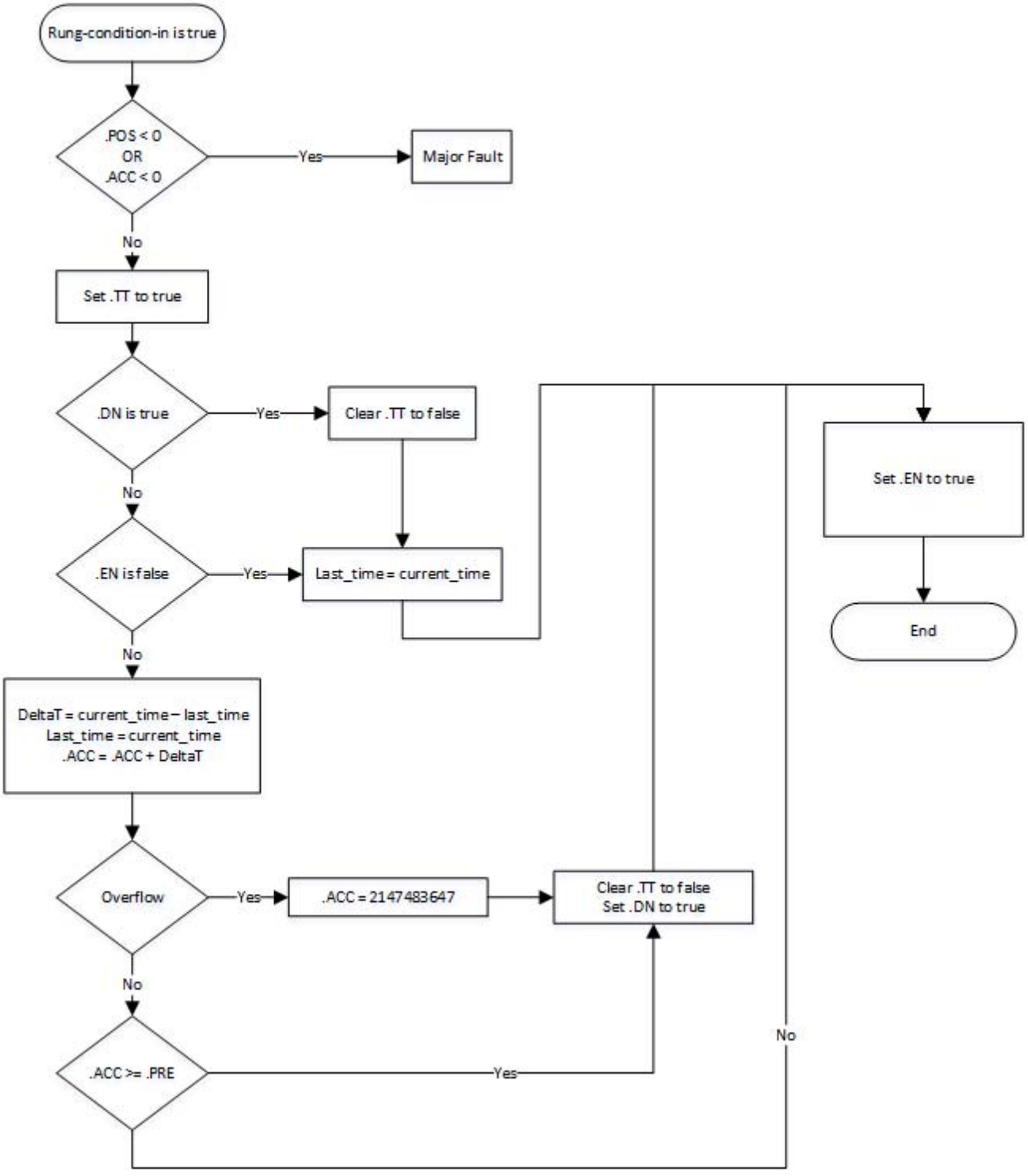
배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

래더 다이어그램

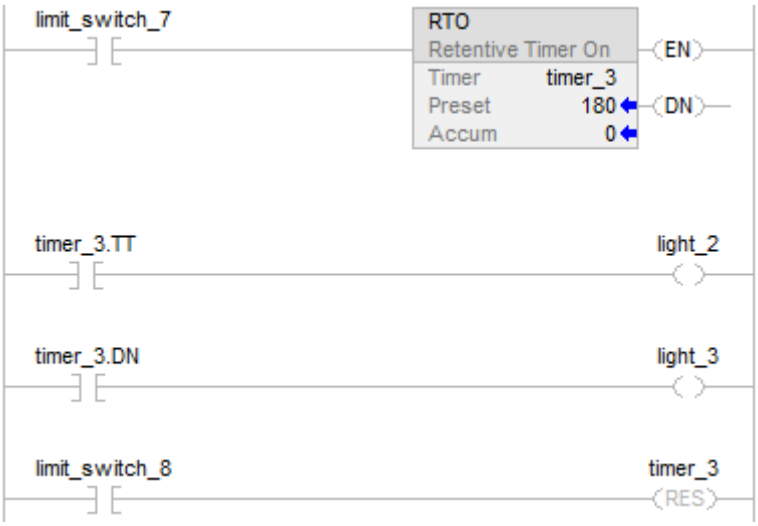
조건/상태	취해진 조치
사전 스캔	.EN 비트가 거짓으로 해제됩니다. .TT 비트가 거짓으로 해제됩니다.
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정 .EN 비트가 거짓으로 해제됩니다. .TT 비트가 거짓으로 해제됩니다.
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 RTO 흐름도(참) 참조하십시오.
사후 스캔	N/A

RTO 흐름도(참)


Rung-condition-in is true
.POS < 0
OR
.ACC < 0
Yes
Major Fault
No
Set .TT to true
.DN is true
Yes
Clear .TT to false
No
.EN is false
Yes
Last_time = current_time
No
DeltaT = current_time – last_time
Last_time = current_time
.ACC = .ACC + DeltaT
Overflow
Yes
.ACC = 2147483647
Clear .TT to false
Set .DN to true
No
.ACC >= .PRE
Yes
No
Set .EN to true
End

예

래더 다이어그램



limit_switch_7 을 설정하면 light_2 가 180 밀리초 동안 켜짐으로 됩니다(timer_3 이 타이밍 중). timer_3.acc 가 180 에 이르면 light_2 는 꺼짐으로, light_3 은 켜짐으로 전환됩니다. timer_3 이 리셋될 때까지 Light_3 은 계속 켜짐입니다. timer_3 이 타이밍하는 동안 limit_switch_7 이 해제되는 경우 light_2 가 꺼짐으로 전환됩니다. limit_switch_7 이 설정된 경우 RES 명령어에 의해 timer_3 이 리셋됩니다(상태 비트 및 .ACC 값이 해제됨).

추가 참조

배열을 통한 인덱스 페이지의 978

리셋(RES) 페이지의 129

꺼짐 지연 타이머(TOF) 페이지의 143

켜짐 지연 타이머(TON) 페이지의 153

타이머 및 카운터 명령어 페이지의 111

리셋 포함 켜짐 적산 타이머(RTOR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

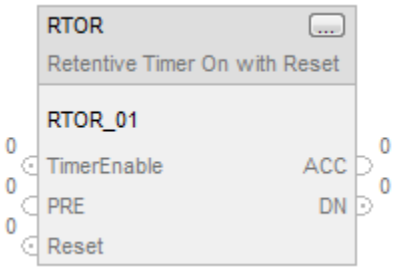
RTOR 명령어는 TimerEnable 이 설정된 시간을 누산하는 적산 타이머입니다.

사용 가능한 언어

래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

평션 블록



ST(스트럭처드 텍스트)

RTOR(RTOR_tag)

피연산자

ST(스트럭처드 텍스트)

변수	유형	형식	설명
RTOR tag	FBD_TIMER	구조	RTOR 구조

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

평션 블록

피연산자	유형	형식	설명
RTOR tag	FBD_TIMER	구조	RTOR 구조

FBD_TIMER 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	선택하지 않으면 명령이 실행되지 않고 출력이 업데이트되지 않습니다. 설정되면 명령어가 실행됩니다. 기본값은 설정 상태입니다.
TimerEnable	BOOL	설정된 경우 타이머가 실행되어 시간을 누산하도록 타이머를 활성화합니다. 기본값은 해제 상태입니다.
PRE	DINT	타이머 사전 설정 값. 타이밍이 완료되기 전 ACC 가 도달해야 하는 1 밀리초 단위의 값입니다. 유효하지 않을 경우 명령어에서 Status 에 해당하는 비트를 설정하고 타이머가 실행되지 않습니다. 유효값 = 0 ~ 양의 최대 정수
Reset	BOOL	타이머 리셋 요청. 설정된 경우 타이머가 리셋됩니다. Reset 입력 파라미터가 설정된 경우 명령어에 의해 EN, TT, DN 이 해제되고 ACC = 0 으로 설정됩니다.

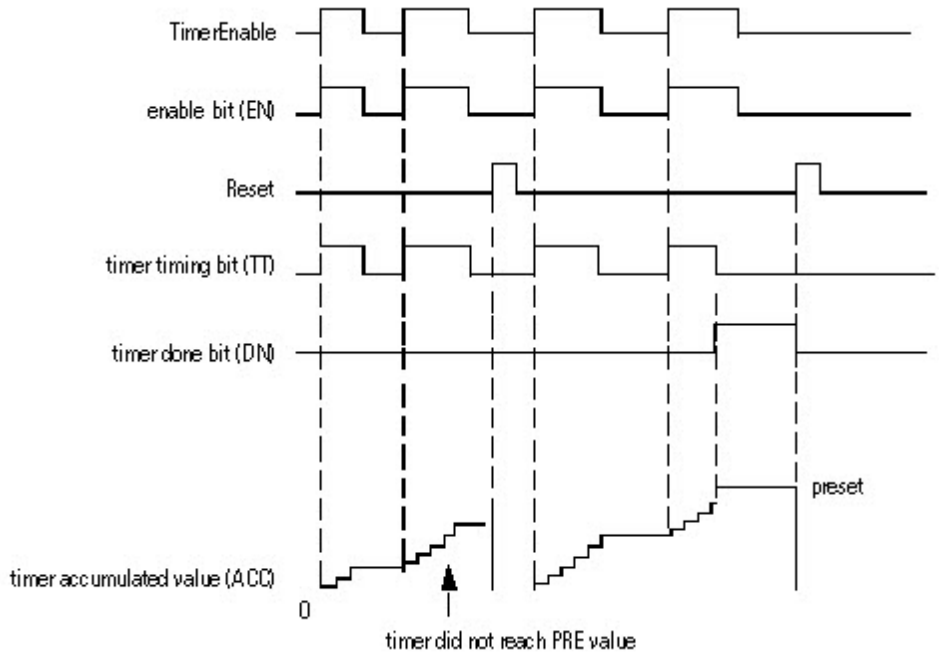
출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어에 의해 유효한 결과가 산출됩니다.
ACC	DINT	밀리초 단위 누산 시간. TimerEnable 입력이 해제된 동안에도 이 값은 그대로 유지됩니다.
EN	BOOL	타이머 활성화 출력. 타이머 명령어가 활성화되었음을 나타냅니다.
TT	BOOL	타이머 타이밍 출력. 설정된 경우 타이밍 작업이 진행 중입니다.
DN	BOOL	타이밍 완료 출력. 누산 시간이 사전 설정 값 이상임을 나타냅니다.
Status	DINT	펑션 블록의 상태.

InstructFault (Status.0)	BOOL	명령어에서 다음 실행 에러 중 하나가 감지됩니다. 마이너 또는 메이저 컨트롤러 에러가 아닙니다. 나머지 상태 비트를 확인하여 발생한 에러를 판단합니다.
PresetInv (Status.1)	BOOL	사전 설정 값이 유효하지 않습니다.

설명

RTOR 명령어는 거짓이 되는 시점에 이르기까지 시간을 누산합니다. RTOR 명령어가 거짓일 경우 ACC 값이 유지됩니다. .ACC 값을 지우려면 Reset 입력을 사용해야 합니다.

시간 기준은 항상 1 밀리초입니다. 예를 들어 2 초 타이머의 경우 PRE 값으로 2000 을 입력합니다.



명령어를 리셋하려면 Reset 입력 파라미터를 설정합니다. Reset 이 설정되었을 때 TimerEnable 이 설정되면 Reset 이 해제될 때 RTOR 명령어에 의해 타이밍이 다시 시작됩니다.

타이머 실행 방식

타이머는 현재 시간에서 타이머의 마지막 스캔 시간을 빼는 식으로 실행됩니다.

- ACC = ACC + (current_time - last_time_scanned)
- ACC 를 업데이트한 후에 타이머는 last_time_scanned = current_time 으로 설정됩니다. 다음 스캔에 대비하여 준비됩니다.

중요:	타이머가 실행되는 동안 최소한 69 분마다 타이머를 스캔해야 합니다. 그렇지 않을 경우 ACC 값이 정확하지 않습니다.

last_time_scanned 값의 범위는 최대 69 분입니다. 69 분 이내에 타이머를 스캔하지 않을 경우 타이머 계산이 롤오버됩니다. 이 경우 ACC 값이 정확하지 않습니다.

다음에서 타이머를 설정한 경우 타이머가 실행되는 동안 69 분 이내에 타이머를 스캔하십시오.

- 서브루틴
- JMP 및 LBL 명령어 간 코드 섹션
- SFC(Sequential Function Chart, 순차 평션 차트)
- 이벤트 또는 주기 태스크
- 단계 상태 루틴

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

펑션 블록

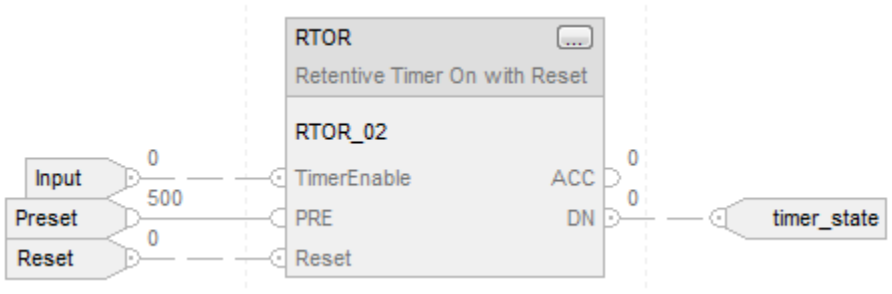
조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 명령어가 실행됩니다. Reset 입력 파라미터가 설정된 경우 명령어에 의해 EN, TT, DN 이 해제되고 ACC = 0 으로 설정됩니다.
명령어 최초 실행	EN, TT, DN 이 거짓으로 해제됩니다. 명령어가 실행됩니다.
명령어 최초 스캔	N/A
사후 스캔	EnableIn 과 EnableOut 이 거짓으로 해제됩니다.

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	펑션 블록 표의 사전 스캔을 참조하십시오.
정상 실행	펑션 블록 표에서 Tag.EnableIn 이 참임을 참조하십시오.
사후 스캔	펑션 블록 표에서 사후 스캔을 참조하십시오.

예

펑션 블록



ST(스트럭쳐드 텍스트)

```
RTOR_01.PRE := 500;

RTOR_01.Reset := Reset;

RTOR_01.TimerEnable := Input;

RTOR(RTOR_01);

timer_state := RTOR_01.DN;
```

추가 참조

공통 속성 페이지의 963

적산 시간 온(RTO) 페이지의 132

리셋(RES) 페이지의 129

ST(스트럭쳐드 텍스트) 구문 페이지의 997

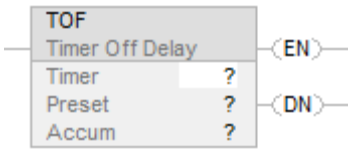
꺼짐 지연 타이머(TOF)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

TOF 명령어는 명령어가 활성화되는 시간을 누산하는 비적산 타이머입니다(렁-입력-조건이 거짓임).

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Timer	TIMER	태그	타이머 구조
Preset	DINT	즉시	Timer.PRE 값.
Accum	DINT	즉시	Timer.ACC 값.

TIMER 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트에는 명령어가 마지막으로 실행되었을 때 렁-입력-조건이 들어 있습니다.
.TT	BOOL	타이밍 비트는 설정된 경우 타이밍 작업이 진행 중임을 나타냅니다.
.DN	BOOL	완료 비트는 해제되면 타이밍 작업이 완료(또는 일시 중지)되었음을 나타냅니다.
.PRE	DINT	사전 설정 값으로 명령어가 완료된 것으로 나타나기 전에 누산 값이 도달해야 하는 값(1 밀리초 단위)을 지정합니다.
.ACC	DINT	누산 값은 TOF 명령어가 활성화된 이후로 경과한 시간을 밀리초로 지정합니다.

설명

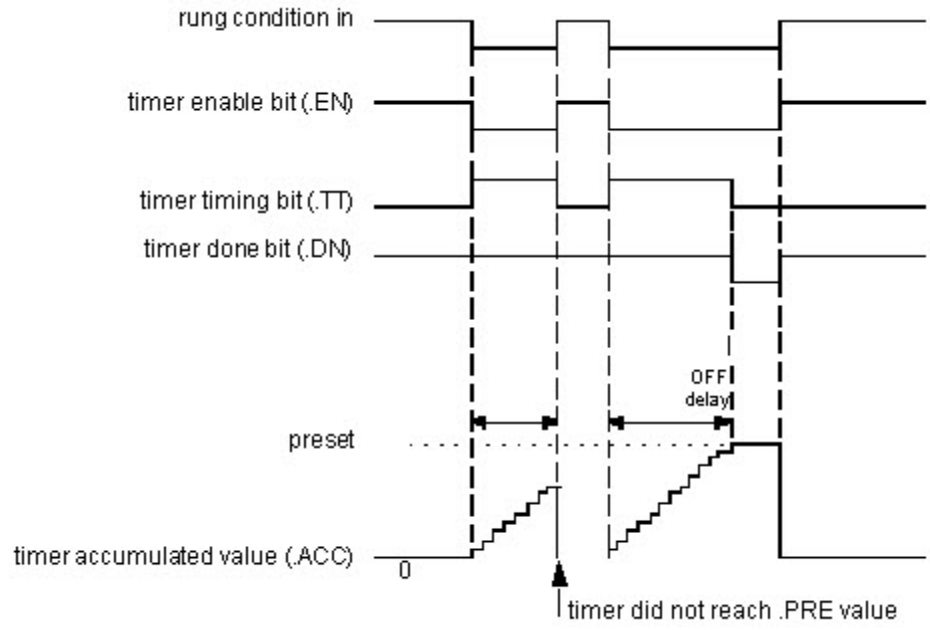
TOF 명령어는 다음 시점에 이르는 시간을 누산합니다.

- 타이머가 비활성화됨
- 타이머 완료

시간 기준은 항상 1 밀리초입니다. 예를 들어 2 초 타이머의 경우 .PRE 값으로 2000 을 입력합니다.

타이머는 타이머가 완료할 때 .DN 비트를 거짓으로 해제합니다.

활성화되었을 때 .DN 비트를 거짓으로 해제하여 타이밍을 일시 중단하고 다시 참으로 설정하여 타이밍을 다시 시작할 수 있습니다.



타이머 실행 방식

타이머는 현재 시간에서 타이머의 마지막 스캔 시간을 빼는 식으로 실행됩니다.

ACC = ACC + (current_time - last_time_scanned)

ACC 를 업데이트한 후에는 타이머가 last_time_scanned = current_time 으로 설정되고, 다음 스캔에 대비하여 준비됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
.PRE < 0	4	34
.ACC < 0	4	34

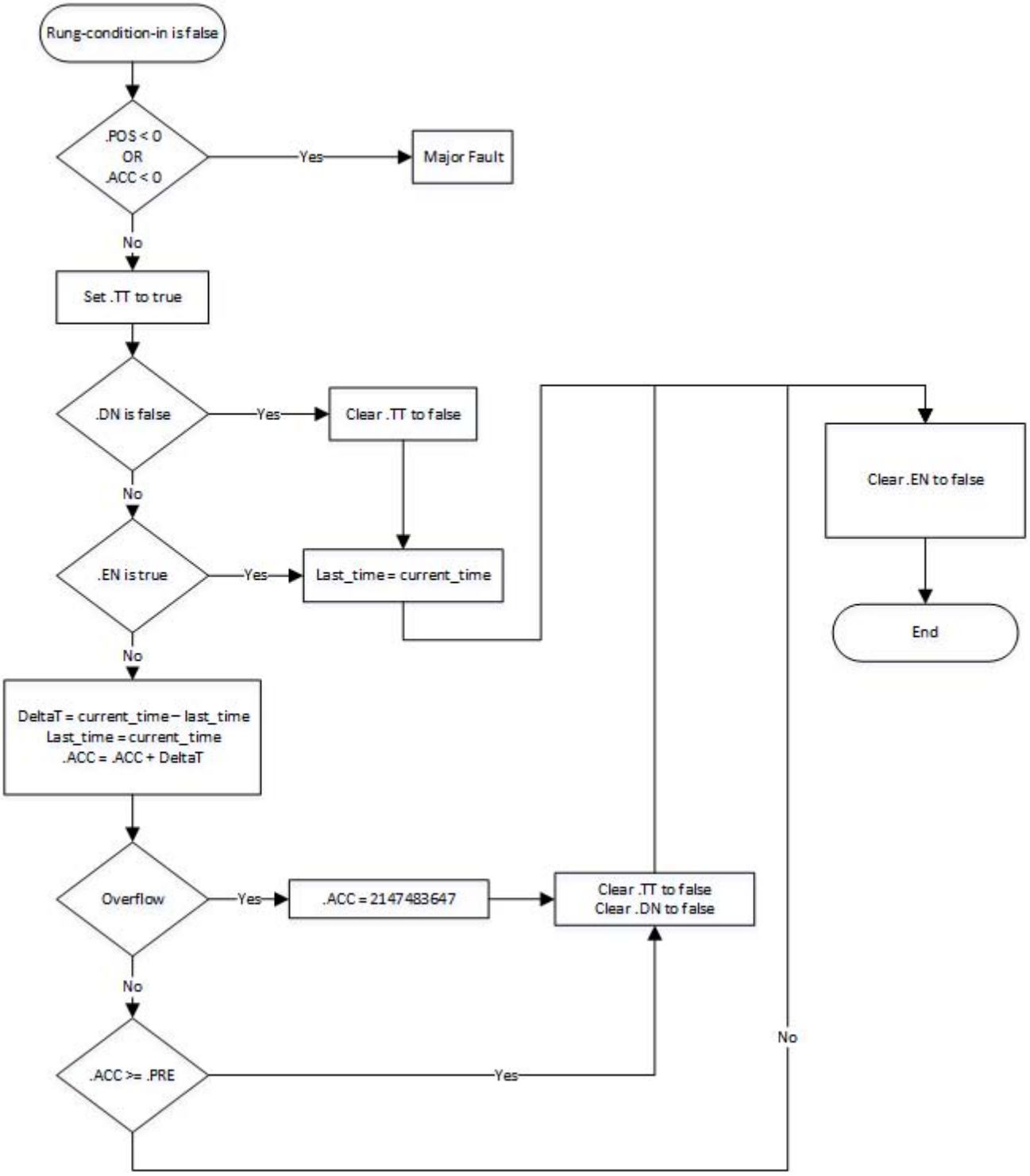
배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

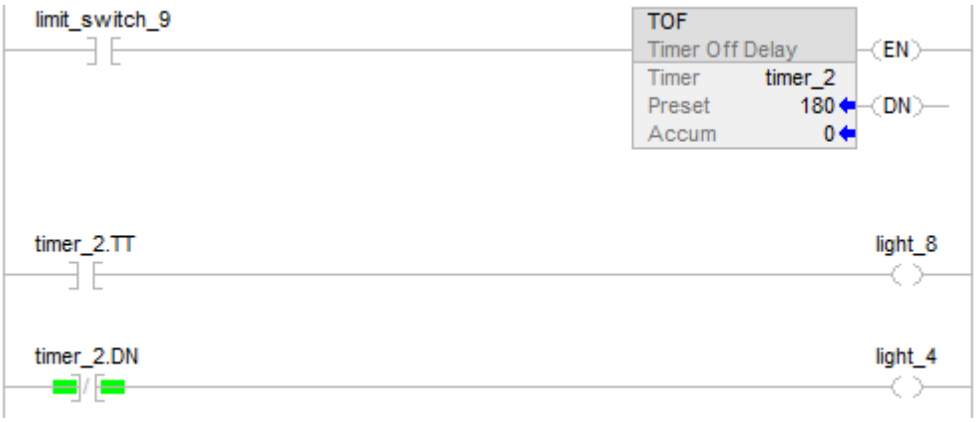
조건/상태	취해진 조치
사전 스캔	.EN 비트가 거짓으로 해제됩니다. .TT 비트가 거짓으로 해제됩니다. .DN 비트가 거짓으로 해제됩니다. .ACC 값이 .PRE 값과 일치하도록 설정됩니다.
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정 TOF 흐름도(거짓) 참조하십시오.
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 .EN 비트가 참으로 설정됩니다. .TT 비트가 거짓으로 해제됩니다. .DN 비트가 참으로 설정됩니다. .ACC 값이 0 으로 해제됩니다.
사후 스캔	.EN 비트가 거짓으로 해제됩니다. .TT 비트가 거짓으로 해제됩니다. .DN 비트가 거짓으로 해제됩니다. .ACC 값이 .PRE 값과 일치하도록 설정됩니다.

TOF 흐름도(거짓)


Rung-condition-in is false
.POS < 0
OR
.ACC < 0
Yes
Major Fault
No
Set .TT to true
.DN is false
Yes
Clear .TT to false
No
.EN is true
Yes
Last_time = current_time
No
DeltaT = current_time – last_time
Last_time = current_time
.ACC = .ACC + DeltaT
Overflow
Yes
.ACC = 2147483647
Clear .TT to false
Clear .DN to false
No
.ACC >= .PRE
Yes
No
Clear .EN to false
End

예

래더 다이어그램



limit_switch_9 가 해제되면 light_8 이 180 밀리초 동안 켜짐으로 됩니다(timer_2 가 타이밍 중). timer_2.acc 가 180 에 도달할 때 light_8 은 꺼짐으로, light_4 는 켜짐으로 됩니다. TOF 명령어가 활성화될 때까지 Light_4 는 계속 켜짐을 유지합니다. timer_2 가 타이밍하는 동안 limit_switch_9 가 참인 경우 light_8 이 꺼짐으로 됩니다.

추가 참조

타이머 및 카운터 명령어 페이지의 111

배열을 통한 인덱스 페이지의 978

리셋 포함 꺼짐 지연 타이머(TOFR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

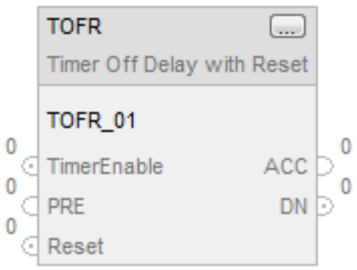
TOFR 명령어는 TimerEnable 이 해제된 시간을 누산하는 비적산 타이머입니다.

사용 가능한 언어

래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

평션 블록



ST(스트럭처드 텍스트)

TOFR(TOFR_tag)

피연산자

ST(스트럭처드 텍스트)

변수	유형	형식	설명
TOFR tag	FBD_TIMER	구조	TOFR 구조

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 ST(스트럭처드 텍스트) 구문을 참조하십시오.

평션 블록

피연산자	유형	형식	설명
TOFR tag	FBD_TIMER	구조	TOFR 구조

FBD_TIMER 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	선택하지 않으면 명령이 실행되지 않고 출력이 업데이트되지 않습니다. 설정되면 명령어가 실행됩니다. 기본값은 설정 상태입니다.
TimerEnable	BOOL	해제된 경우 타이머가 실행되어 시간을 누산하도록 타이머를 활성화하는 파라미터입니다. 기본값은 해제 상태입니다.

PRE	DINT	타이머 사전 설정 값. 타이밍이 완료되기 전 ACC 가 도달해야 하는 1 밀리초 단위의 값입니다. 유효하지 않을 경우 명령어에서 Status 에 해당하는 비트를 설정하고 타이머가 실행되지 않습니다. 유효값 = 0 ~ 양의 최대 정수
Reset	BOOL	타이머 리셋 요청. 설정된 경우 타이머가 리셋됩니다. 기본값은 해제 상태입니다. Reset 입력 파라미터가 설정된 경우 명령어에 의해 EN, TT, DN 이 해제되고 ACC = PRE 로 설정됩니다. TOF 명령어 실행 후 RES 명령어를 사용하는 것과 다릅니다.

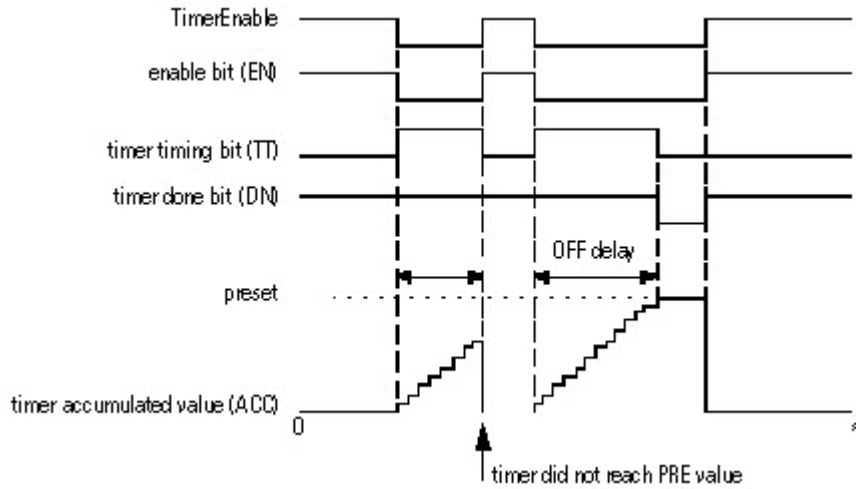
출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어에 의해 유효한 결과가 산출됩니다.
ACC	BOOL	밀리초 단위 누산 시간.
EN	BOOL	타이머 활성화 출력. 타이머 명령어가 활성화되었음을 나타냅니다.
TT	BOOL	타이머 타이밍 출력. 설정된 경우 타이밍 작업이 진행 중입니다.
DN	BOOL	타이밍 완료 출력. 누산 시간이 사전 설정 값 이상임을 나타냅니다.
Status	DINT	펑션 블록의 상태.
InstructFault (Status.0)	BOOL	명령어에서 다음 실행 에러 중 하나가 감지됩니다. 마이너 또는 메이저 컨트롤러 에러가 아닙니다. 나머지 상태 비트를 확인하여 발생한 에러를 판단합니다.
PresetInv (Status.1)	BOOL	사전 설정 값이 유효하지 않습니다.

설명

참인 경우 TOFR 명령어에 의해 다음 시점까지 시간이 누산됩니다.

- TOFR 명령어가 비활성화됨
- ACC ≥ PRE

시간 기준은 항상 1 밀리초입니다. 예를 들어 2 초 타이머의 경우 PRE 값으로 2000 을 입력합니다.



명령어를 리셋하려면 Reset 입력 파라미터를 설정합니다. Reset 이 참일 때 TimerEnable 이 거짓인 경우 Reset 이 거짓으로 될 때 TOFR 명령어에서 다시 타이밍을 시작하지 않습니다.

타이머 실행 방식

타이머는 현재 시간에서 타이머의 마지막 스캔 시간을 빼는 식으로 실행됩니다.

ACC = ACC + (current_time - last_time_scanned)

ACC 를 업데이트한 후에 타이머는 last_time_scanned = current_time 으로 설정됩니다. 다음 스캔에 대비하여 준비됩니다.

중요: 타이머가 실행되는 동안 최소한 69 분마다 타이머를 스캔해야 합니다. 그렇지 않을 경우 ACC 값이 정확하지 않습니다.

last_time_scanned 값의 범위는 최대 69 분입니다. 69 분 이내에 타이머를 스캔하지 않을 경우 타이머 계산이 롤오버됩니다. 이 경우 ACC 값이 정확하지 않습니다.

다음에서 타이머를 설정한 경우 타이머가 실행되는 동안 69 분 이내에 타이머를 스캔하십시오.

- 서브루틴

- JMP 및 LBL 명령어 간 코드 섹션
- SFC(Sequential Function Chart, 순차 펑션 차트)
- 이벤트 또는 주기 태스크
- 단계 상태 루틴

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

펑션 블록

조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag. EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag. EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 명령어의 주 알고리즘이 실행되고 출력이 업데이트됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	EN, TT, DN 이 해제되고 ACC 값이 수정되지 않습니다.
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

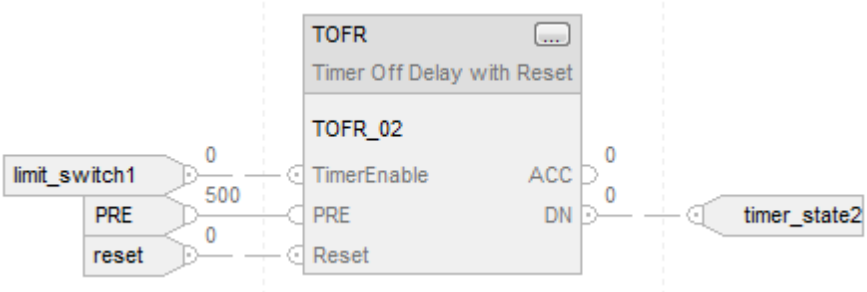
ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	펑션 블록 표의 사전 스캔을 참조하십시오.
정상 실행	펑션 블록 표에서 Tag.EnableIn 이 참임을 참조하십시오.
사후 스캔	펑션 블록 표에서 사후 스캔을 참조하십시오.

예

limit_switch1 이 해제된 후 스캔할 때마다 TOFR 명령어는 ACC 값이 PRE 값에 도달할 때까지 경과된 시간만큼 ACC 값을 증가시킵니다. ACC ≥ PRE 일 경우 DN 파라미터가 해제되고 timer_state2 가 설정됩니다.

펑션 블록



ST(스트럭처드 텍스트)

```
TOFR_01.PRE := 500;

TOFR_01.Reset := Reset;

TOFR_01.TimerEnable := Input;

TOFR(TOFR_01);

timer_state := TOFR_01.DN;
```

추가 참조

공통 속성 페이지의 963

ST(스트럭처드 텍스트) 구문 페이지의 997

켜짐 지연 타이머(TON)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

TON 명령어는 명령어가 활성화된 시간을 누산하는 비적산 타이머입니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Timer	TIMER	태그	타이머 구조
Preset	DINT	즉시	Timer.PRE 값.
Accum	DINT	즉시	Timer.ACC 값.

TIMER 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트에는 명령어가 마지막으로 실행되었을 때 렁-입력-조건이 들어 있습니다.
.TT	BOOL	타이밍 비트는 설정된 경우 타이밍 작업이 진행 중임을 나타냅니다.

.DN	BOOL	완료 비트는 설정된 경우 타이밍 작업이 완료(또는 일시 중지)되었음을 나타냅니다.
.PRE	DINT	사전 설정 값으로 명령어가 완료된 것으로 나타나기 전에 누산 값이 도달해야 하는 값(1 밀리초 단위)을 지정합니다.
.ACC	DINT	누산 값은 TON 명령어가 활성화된 이후로 경과한 시간을 밀리초로 지정합니다.

설명

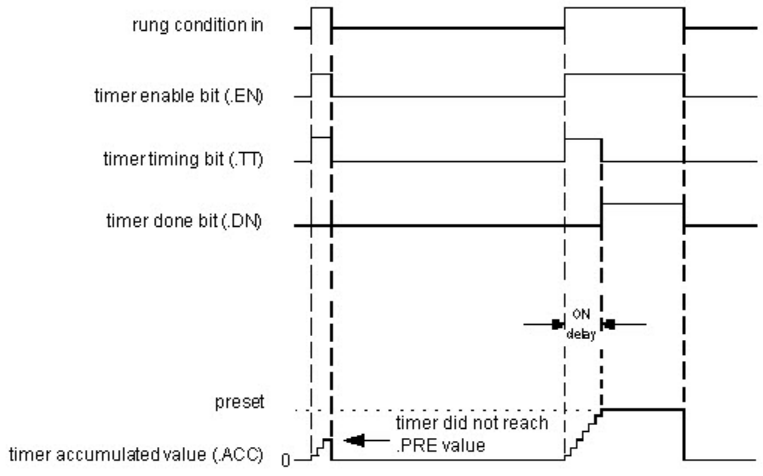
TON 명령어는 활성화된 때부터 다음 시점까지의 시간을 누산합니다.

- 타이머가 비활성화됨
- 타이머 완료

시간 기준은 항상 1 밀리초입니다. 예를 들어 2 초 타이머의 경우 .PRE 값으로 2000 을 입력합니다.

타이머는 타이머가 완료할 때 .DN 비트를 참으로 설정합니다.

활성화되었을 때 .DN 비트를 참으로 설정하여 타이밍을 일시 중단한 후에 다시 거짓으로 해제하여 타이밍을 다시 시작할 수 있습니다.



타이머 실행 방식

타이머는 현재 시간에서 타이머의 마지막 스캔 시간을 빼는 식으로 실행됩니다.

ACC = ACC + (current_time - last_time_scanned)

ACC 를 업데이트한 후에는 타이머가 last_time_scanned = current_time 으로 설정되고, 다음 스캔에 대비하여 준비됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
.PRE < 0	4	34
.ACC < 0	4	34

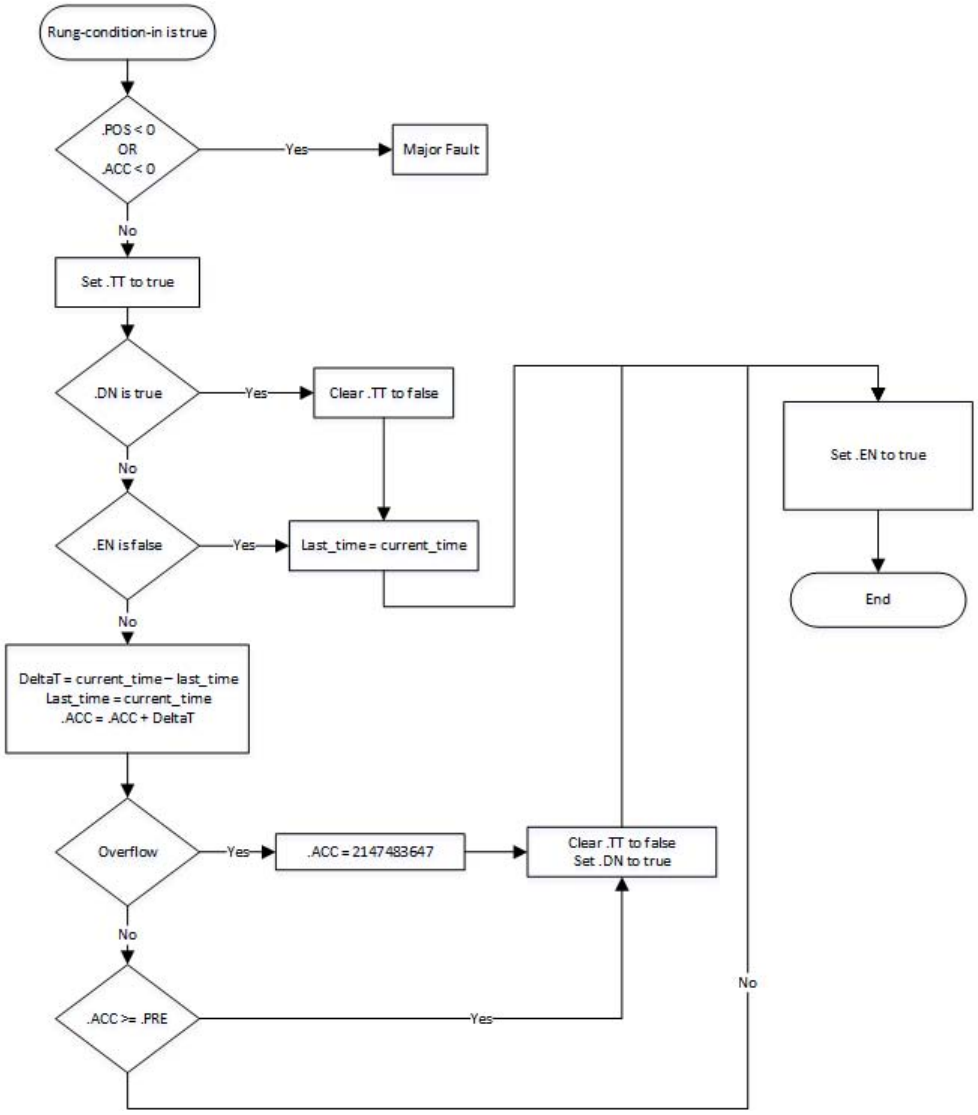
배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

래더 다이어그램

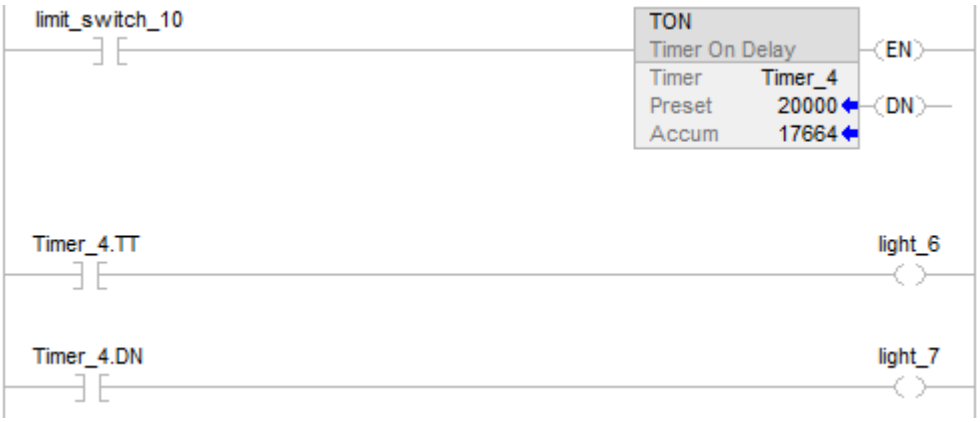
조건/상태	취해진 조치
사전 스캔	.EN 비트가 거짓으로 해제됩니다. .TT 비트가 거짓으로 해제됩니다. .DN 비트가 거짓으로 해제됩니다. .ACC 값이 0 으로 해제됩니다.
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정 .EN 비트가 거짓으로 해제됩니다. .TT 비트가 거짓으로 해제됩니다. .DN 비트가 거짓으로 해제됩니다. .ACC 값이 0 으로 해제됩니다.
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 TON 흐름도(참) 참조하십시오.
사후 스캔	.EN 비트가 거짓으로 해제됩니다. .TT 비트가 거짓으로 해제됩니다. .DN 비트가 거짓으로 해제됩니다. .ACC 값이 0 으로 해제됩니다.

TON 흐름도(참)


Rung-condition-in is true
.POS < 0
OR
.ACC < 0
Yes
Major Fault
No
Set .TT to true
.DN is true
Yes
Clear .TT to false
No
.EN is false
Yes
Last_time = current_time
No
DeltaT = current_time – last_time
Last_time = current_time
.ACC = .ACC + DeltaT
Overflow
Yes
.ACC = 2147483647
Clear .TT to false
Set .DN to true
No
.ACC >= .PRE
Yes
No
Set .EN to true
End

예

래더 다이어그램



limit_switch_10 이 참으로 설정되면 light_6 이 20000 밀리초 동안 켜짐으로 됩니다(Timer_4 가 타이밍 중). Timer_4.acc 가 20000 에 이르면 light_6 은 꺼짐, light_7 은 켜짐으로 됩니다. Timer_4 가 타이밍하는 동안 limit_switch_10 이 거짓으로 해제되는 경우 light_6 은 꺼짐으로 됩니다. limit_switch_10 이 거짓으로 해제되면 Timer_4 상태 비트 및 .ACC 값이 리셋됩니다.

추가 참조

카운터 명령어 페이지의 111

배열을 통한 인덱스 페이지의 978

리셋 포함 켜짐 지연 타이머(TONR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

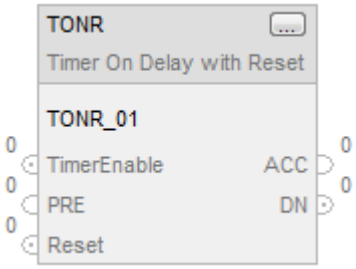
TONR 명령어는 TimerEnable 이 설정된 시간을 누산하는 비적산 타이머입니다.

사용 가능한 언어

래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

평션 블록



ST(스트럭처드 텍스트)

TONR(TONR_tag);

피연산자

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
TONR tag	FBD_TIMER	구조	TONR 구조

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 ST(스트럭처드 텍스트) 구문을 참조하십시오.

평션 블록

피연산자	유형	형식	설명
TONR tag	FBD_TIMER	구조	TONR 구조

FBD_TIMER 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	선택하지 않으면 명령이 실행되지 않고 출력이 업데이트되지 않습니다. 설정되면 명령어가 실행됩니다. 기본값은 설정 상태입니다.
TimerEnable	BOOL	설정된 경우 타이머가 실행되어 시간을 누산하도록 타이머를 활성화합니다. 기본값은 해제 상태입니다.

PRE	DINT	타이머 사전 설정 값. 타이밍이 완료되기 전 ACC 가 도달해야 하는 1 밀리초 단위의 값입니다. 유효하지 않을 경우 명령어에서 Status 에 해당하는 비트를 설정하고 타이머가 실행되지 않습니다. 유효값 = 0 ~ 양의 최대 정수
Reset	BOOL	타이머 리셋 요청. 설정된 경우 타이머가 리셋됩니다. 기본값은 해제 상태입니다. Reset 입력 파라미터가 설정된 경우 명령어에 의해 EN, TT, DN 이 해제되고 ACC = 0 으로 설정됩니다.

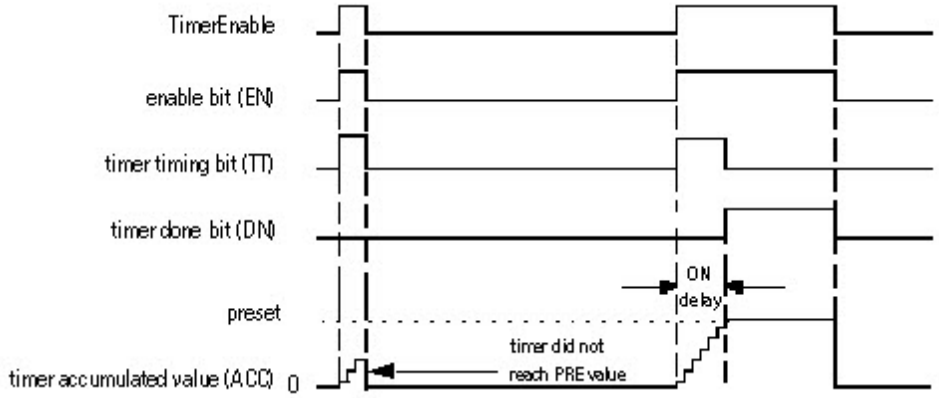
출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어에 의해 유효한 결과가 산출됩니다.
ACC	BOOL	밀리초 단위 누산 시간.
ENF	BOOL	타이머 활성화 출력. 타이머 명령어가 활성화되었음을 나타냅니다.
TT	BOOL	타이머 타이밍 출력. 설정된 경우 타이밍 작업이 진행 중입니다.
DN	BOOL	타이밍 완료 출력. 누산 시간이 사전 설정 값 이상임을 나타냅니다.
Status	DINT	펑션 블록의 상태.
InstructFault (Status.0)	BOOL	명령어에서 다음 실행 에러 중 하나가 감지됩니다. 마이너 또는 메이저 컨트롤러 에러가 아닙니다. 나머지 상태 비트를 확인하여 발생한 에러를 판단합니다.
PresetInv (Status.1)	BOOL	사전 설정 값이 유효하지 않습니다.

설명

참인 경우 TONR 명령어에 의해 다음 시점까지 시간이 누산됩니다.

- TONR 명령어가 비활성화됨
- ACC ≥ PRE

시간 기준은 항상 1 밀리초입니다. 예를 들어 2 초 타이머의 경우 PRE 값으로 2000 을 입력합니다.



명령어를 리셋하려면 Reset 입력 파라미터를 설정합니다. Reset 이 참일 때 TimerEnable 이 설정되면 Reset 이 거짓으로 되면 TONR 명령어에 의해 다시 타이밍이 시작됩니다.

타이머 실행 방식

타이머는 현재 시간에서 타이머의 마지막 스캔 시간을 빼는 식으로 실행됩니다.

- ACC = ACC + (current_time - last_time_scanned)

ACC 를 업데이트한 후에 타이머는 last_time_scanned = current_time 으로 설정됩니다. 다음 스캔에 대비하여 준비됩니다.

중요: 타이머가 실행되는 동안 최소한 69 분마다 타이머를 스캔해야 합니다. 그렇지 않은 경우 ACC 값이 맞지 않습니다.

last_time_scanned 값의 범위는 최대 69 분입니다. 69 분 이내에 타이머를 스캔하지 않을 경우 타이머 계산이 롤오버됩니다. 이 경우 ACC 값이 정확하지 않습니다.

다음에서 타이머를 설정한 경우 타이머가 실행되는 동안 69 분 이내에 타이머를 스캔하십시오.

- 서브루틴
- JMP 및 LBL 명령어 간 코드 섹션
- SFC(Sequential Function Chart, 순차 평션 차트)
- 이벤트 또는 주기 태스크
- 단계 상태 루틴

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

펑션 블록

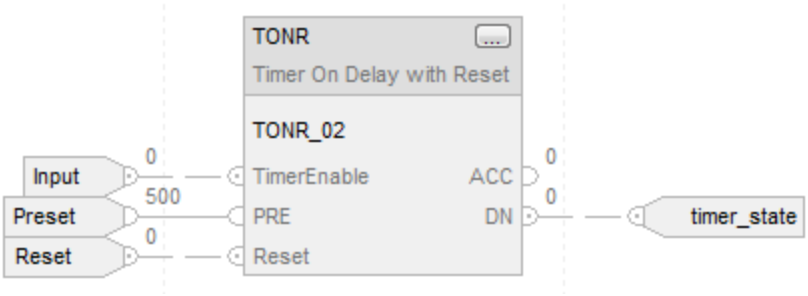
조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 명령어의 주 알고리즘이 실행되고 출력이 업데이트됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	EN, TT, DN 이 해제되고 ACC 값이 0 으로 설정됩니다.
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	펑션 블록 표의 사전 스캔을 참조하십시오.
정상 실행	펑션 블록 표에서 Tag.EnableIn 이 참임을 참조하십시오.
사후 스캔	펑션 블록 표에서 사후 스캔을 참조하십시오.

예

평션 블록



ST(스트럭쳐드 텍스트)

```
TONR_01.PRE := 500;

TONR_01.Reset := Reset;

TONR_01.TimerEnable := Input;

TONR(TONR_01);

timer_state := TONR_01.DN;
```

추가 참조

공통 속성 페이지의 963

켜짐 지연 타이머(TON) 페이지의 153

리셋(RES) 페이지의 129

ST(스트럭쳐드 텍스트) 구문 페이지의 997

입력/출력

입력/출력 명령어

입력/출력 명령어는 컨트롤러의 데이터를 읽기/쓰기하거나 다른 네트워크의 다른 모듈의 데이터 블록을 읽기/쓰기 합니다.

사용 가능한 명령어

래더 다이어그램과 ST(스트럭처드 텍스트)

MSG	GSV	SSV	IOT

평션 블록

사용할 수 없음

실행할 작업:	사용할 명령어
모듈 간 데이터 전송	MSG
컨트롤러 상태 정보 가져오기	GSV
컨트롤러 상태 정보 설정	SSV
I/O 모듈 또는 로직의 특정 포인트에서 소비 컨트롤러에 출력 값 보내기 다른 컨트롤러에서 이벤트 태스크 트리거	IOT

추가 참조

통신 세부 정보 지정 페이지의 197

CIP 메세지 지정하기 페이지의 318

메세지 유형 선택 페이지의 298

MSG 구성 예 페이지의 177

컨트롤러 메모리 정보 알아내기 페이지의 217

메세지(MSG)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

MSG 명령어는 비동기적으로 데이터 블록을 읽거나 네트워크의 다른 모듈에 씁니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

MSG(MessageControl);

피연산자

래더 다이어그램

피연산자	유형	형식	설명
Message	MSG	태그	메세지 구조

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Message	MSG	태그	메세지 구조

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

MESSAGE 구조

중요: 상태 비트를 2 회 이상 확인하는 경우
로직 내 두 군데 이상 부분에서 비트를 확인하는 경우 비트 복사본을 사용합니다. 그렇지 않을 경우 스캔 도중에 비트가 변경될 수 있어 로직이 예상대로 작동하지 않습니다.
사본을 만드는 한 가지 방법은 FLAGS 워드를 사용하는 방법입니다. FLAGS 워드를 다른 태그에 복사하고 복사본에서 비트를 확인합니다.

중요: MSG 명령어에서 다음 비트를 변경하지 마십시오.
- DN
- EN
- ER
- EW
- ST

단독 또는 FLAGS 워드의 일부로 사용된 경우 비트를 변경하지 마십시오. 변경할 경우 컨트롤러에 복구할 수 없는 폴트가 발생할 수 있습니다. 컨트롤러에 복구할 수 없는 폴트가 발생할 경우 컨트롤러 메모리에서 프로젝트가 지워집니다.

니모닉	데이터 유형	설명	
.FLAGS	INT	.FLAGS 구성원은 16 비트로 된 한 워드의 상태 구성원(비트)에 액세스하게 해줍니다.	
		해당 비트 2	**해당하는 구성원 .EW**
		4	.ER
		5	.DN
		6	.ST
		7	.EN
		8	.TO
		9	.EN_CC
		중요: FLAGS 구성원의 EW, ER, DN 또는 ST 비트를 변경하지 마십시오. 예를 들어 전체 FLAGS 워드를 지우지 마십시오. 컨트롤러는 변경 내용을 무시하고 내부 저장된 비트 값을 사용합니다.	

.ERR	INT	.ER 비트가 설정된 경우 에러 코드 워드는 MSG 명령어의 에러 코드를 나타냅니다.
.EXERR	INT	확장 에러 코드 워드는 일부 에러 코드에 대한 추가적인 에러 코드 정보를 지정합니다.
.REQ_LEN	INT	요청된 길이는 메세지 명령어에 의해 전송 시도되는 워드 수를 지정합니다.
.DN_LEN	INT	완료 길이는 실제 전송된 워드 수를 나타냅니다.
.EW	BOOL	컨트롤러에서 메세지 요청이 큐에 들어간 것으로 감지될 때 활성화 대기 비트가 설정됩니다. .ST 비트가 설정되면 컨트롤러에서 .EW 비트가 리셋됩니다. **중요:** EW 비트를 변경하지 마십시오. 컨트롤러는 변경 내용을 무시하고 내부 저장된 비트 값을 사용합니다.
.ER	BOOL	컨트롤러에서 전송 실패가 감지될 경우 에러 비트가 설정됩니다. 다음에 EnableIn 이 거짓에서 참으로 될 때 .ER 비트가 리셋됩니다. **중요:** ER 비트를 변경하지 마십시오. 컨트롤러는 변경 내용을 무시하고 내부 저장된 비트 값을 사용합니다.
.DN	BOOL	마지막 메세지 패킷의 전송이 완료될 때 완료 비트가 설정됩니다. 다음에 EnableIn 이 거짓에서 참으로 될 때 .DN 비트가 리셋됩니다. **중요:** DN 비트를 변경하지 마십시오. 컨트롤러는 변경 내용을 무시하고 내부 저장된 비트 값을 사용합니다.
.ST	BOOL	컨트롤러가 MSG 명령어 실행을 시작하면 시작 비트가 설정됩니다. .DN 비트 또는 .ER 비트가 설정되면 .ST 비트가 리셋됩니다. **중요:** ST 비트를 변경하지 마십시오. 컨트롤러는 변경 내용을 무시하고 내부 저장된 비트 값을 사용합니다.

.EN	BOOL	EnableIn 이 참으로 되고 .DN 비트 또는 .ER 비트가 설정되고 EnableIn 이 거짓으로 될 때까지 계속 설정되는 경우 활성화 비트가 설정됩니다. EnableIn 이 거짓으로 되지만 .DN 비트 및 .ER 비트가 해제된 경우 .EN 비트는 계속 설정되어 있습니다. **중요:** EN 비트를 변경하지 마십시오. 컨트롤러는 변경 내용을 무시하고 내부 저장된 비트 값을 사용합니다.
.TO	BOOL	.TO 비트를 수동 설정하는 경우 컨트롤러에서 메세지 처리를 중단하고 .ER 비트를 설정합니다.
.EN_CC	BOOL	활성화 캐시 비트는 MSG 연결의 관리 방법을 결정합니다. 컨트롤러가 연결을 유지하길 원하는 경우(예: 동일한 MSG 명령어를 여러 번 반복 실행하는 경우) .EN_CC 비트를 설정합니다. 좀처럼 MSG 명령어를 실행하지 않고 다른 컨트롤러 연결이 필요한 경우 .EN_CC 비트를 해제합니다. MSG 명령어 연결이 끊어지면 .EN_CC 비트가 설정된 경우에도 시리얼 포트가 캐싱되지 않습니다.
.ERR_SRC	SINT	메세지 구성 대화 상자에 에러 경로를 표시합니다.
.DestinationLink	INT	DH+ 또는 소스 ID 포함 CIP 메세지의 대상 링크를 변경하려면 이 구성원을 필요한 값으로 설정합니다.
.DestinationNode	INT	DH+ 또는 소스 ID 포함 CIP 메세지의 대상 노드를 변경하려면 이 구성원을 필요한 값으로 설정합니다.
.SourceLink	INT	DH+ 또는 소스 ID 포함 CIP 메세지의 소스 링크를 변경하려면 이 구성원을 필요한 값으로 설정합니다.
.Class	INT	CIP 일반 메세지의 Class 파라미터를 변경하려면 이 구성원을 필요한 값으로 설정합니다.

<table>
<tr><td>.Attribute</td><td>INT</td><td colspan="2">CIP 일반 메세지의 Attribute 파라미터를 변경하려면 이 구성원을 필요한 값으로 설정합니다.</td></tr>
<tr><td>.Instance</td><td>DINT</td><td colspan="2">CIP 일반 메세지의 Instance 파라미터를 변경하려면 이 구성원을 필요한 값으로 설정합니다.</td></tr>
<tr><td rowspan="4">.LocalIndex</td><td rowspan="4">DINT</td><td colspan="2">별표[*]를 사용하여 로컬 배열의 요소 번호를 지정하는 경우 LocalIndex 의 요소 번호를 사용합니다. 요소 번호를 변경하려면 이 구성원을 필요한 값으로 설정합니다.</td></tr>
<tr><td>메세지:</td><td>로컬 배열:</td></tr>
<tr><td>데이터 읽기</td><td>대상 요소</td></tr>
<tr><td>데이터 쓰기</td><td>소스 요소</td></tr>
<tr><td>.Channel</td><td>SINT</td><td colspan="2">1756-DHRIO 모듈의 다른 채널을 통해 메세지를 발송하려면 이 구성원을 필요한 값으로 설정합니다. ASCII 문자 A 또는 B 를 사용합니다.</td></tr>
<tr><td>.Rack</td><td>SINT</td><td colspan="2">블록 전송 메세지의 랙 번호를 변경하려면 이 구성원을 필요한 랙 번호(8 진수)로 설정합니다.</td></tr>
<tr><td>.Group</td><td>SINT</td><td colspan="2">블록 전송 메세지의 그룹 번호를 변경하려면 이 구성원을 필요한 그룹 번호(8 진수)로 설정합니다.</td></tr>
<tr><td rowspan="4">.Slot</td><td rowspan="4">SINT</td><td colspan="2">블록 전송 메세지의 슬롯 번호를 변경하려면 이 구성원을 필요한 슬롯 번호로 설정합니다.</td></tr>
<tr><td>메세지가 통과하는 네트워크:</td><td>슬롯 번호 지정 형식:</td></tr>
<tr><td>범용 원격 I/O</td><td>8 진수</td></tr>
<tr><td>ControlNet</td><td>10 진수(0-15)</td></tr>
</table>

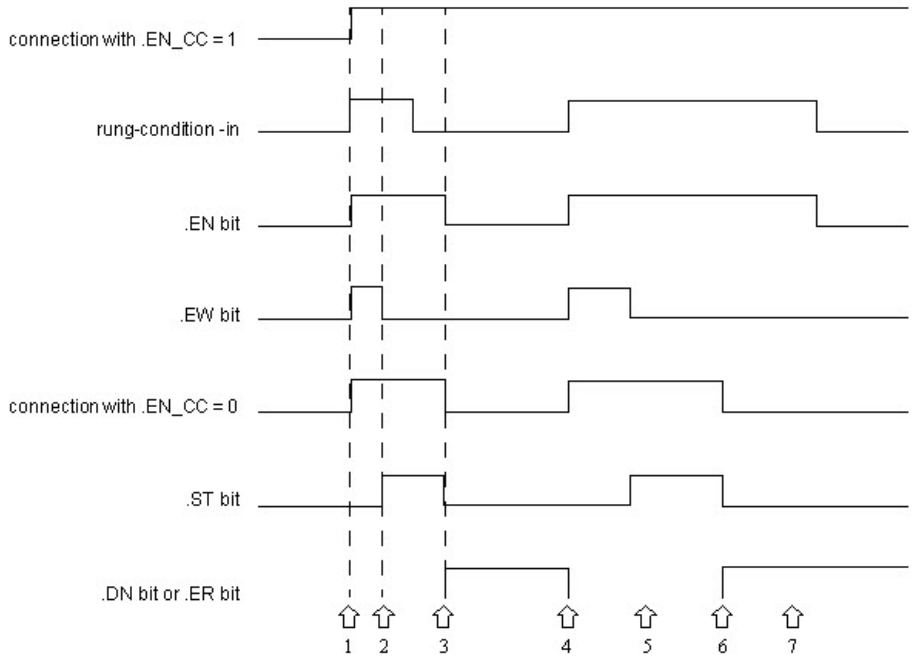
<table>
<tr><td>.Path</td><td>STRING</td><td>다른 컨트롤러에 메세지를 전송하려면 이 구성원을 새 경로로 설정합니다.
16 진수 값 경로를 입력합니다.
쉼표[,] 생략
예를 들어 1, 0, 2, 42, 1, 3, 경로에 대해 $01$00$02$2A$01$03 을 입력합니다.
장치를 탐색하고 새 문자열 전체 또는 일부를 자동으로 생성하려는 경우 문자열 태그를 마우스 오른쪽 단추로 클릭하고 메세지 경로 편집기로 이동(Go to Message Path Editor)을 선택합니다.</td></tr>
<tr><td>.RemoteIndex</td><td>DINT</td><td>별표[*]를 사용하여 원격 배열의 요소 번호를 지정하는 경우 RemoteIndex 의 요소 번호를 사용합니다. 요소 번호를 변경하려면 이 구성원을 필요한 값으로 설정합니다.
<table>
<tr><th>메세지</th><th>원격 배열</th></tr>
<tr><td>데이터 읽기</td><td>소스 요소</td></tr>
<tr><td>데이터 쓰기</td><td>대상 요소</td></tr>
</table></td></tr>
<tr><td>.RemoteElement</td><td>STRING</td><td>메세지를 보낸 컨트롤러에서 다른 태그 또는 주소를 지정하려면 이 구성원을 필요한 값으로 설정합니다. ASCII 문자 형태의 태그 또는 주소를 입력합니다.
<table>
<tr><th>메세지</th><th>원격 배열</th></tr>
<tr><td>데이터 읽기</td><td>소스 요소</td></tr>
<tr><td>데이터 쓰기</td><td>대상 요소</td></tr>
</table></td></tr>
<tr><td>.UnconnectedTimeout</td><td>DINT</td><td>연결되지 않은 메세지 또는 연결에 대한 타임아웃. 기본값은 30 초입니다.
메세지가 연결 안 됨인 경우 컨트롤러가 UnconnectedTimeout 시간 이내에 응답을 얻지 못하면 ER 비트가 켜짐으로 전환됩니다.
메세지가 연결됨인 경우 컨트롤러가 UnconnectedTimeout 시간 이내에 연결하기에 대한 응답을 얻지 못하면 ER 비트가 켜짐으로 전환됩니다.</td></tr>
</table>

.ConnectionRate	DINT	연결된 후 연결된 메세지에 대한 타임아웃. 다른 장치의 응답에 대한 타임아웃입니다. 이 타임아웃은 연결된 후에만 적용됩니다. 타임아웃 = ConnectionRate x TimeoutMultiplier ConnectionRate 의 기본값은 7.5 초입니다. TimeoutMultiplier 의 기분값은 0 입니다(곱셈 배율 4 에 해당). 연결된 메세지의 기본 타임아웃은 30 초입니다(7.5 초 x 4 = 30 초). 타임아웃을 변경하려면 ConnectionRate 를 변경하고 TimeoutMultiplier 를 기본값 그대로 둡니다.
.TimeoutMultiplier	SINT	

설명

MSG 명령어는 데이터 요소를 전송합니다. 이는 전환 명령어에 해당합니다.

- 래더 다이어그램에서 명령어가 실행될 때마다 EnableIn 이 해제에서 설정으로 전환됩니다.
- 각 요소의 크기는 지정하는 데이터 유형 및 사용하는 메세지 명령 유형에 따라 다릅니다.



위치	설명
1	EnableIn 이 참임 .EN 설정됨 .EW 설정됨 연결이 열려 있음
2	메세지 발송됨 .ST 설정됨 .EW 해제됨
3	메세지 완료 또는 에러 발생, EnableIn 이 거짓임 .DN 또는 .ER 설정됨 .ST 해제됨 연결이 닫힘(.EN_CC = 0 일 경우) .EN 해제됨(EnableIn 이 거짓이므로)
4	EnableIn 이 참이고 .DN 또는 .ER 이 이전에 설정됨 .EN 설정됨 .EW 설정됨 연결이 열려 있음 .DN 또는 .ER 해제됨
5	메세지 발송됨 .ST 설정됨 .EW 해제됨
6	메세지 완료 또는 에러 발생, EnableIn 이 계속 참 .DN 또는 .ER 설정됨 .ST 해제됨 연결이 닫힘(.EN_CC = 0 일 경우)
7	EnableIn 거짓 전환, .DN 또는 .ER 설정됨 .EN 해제됨

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

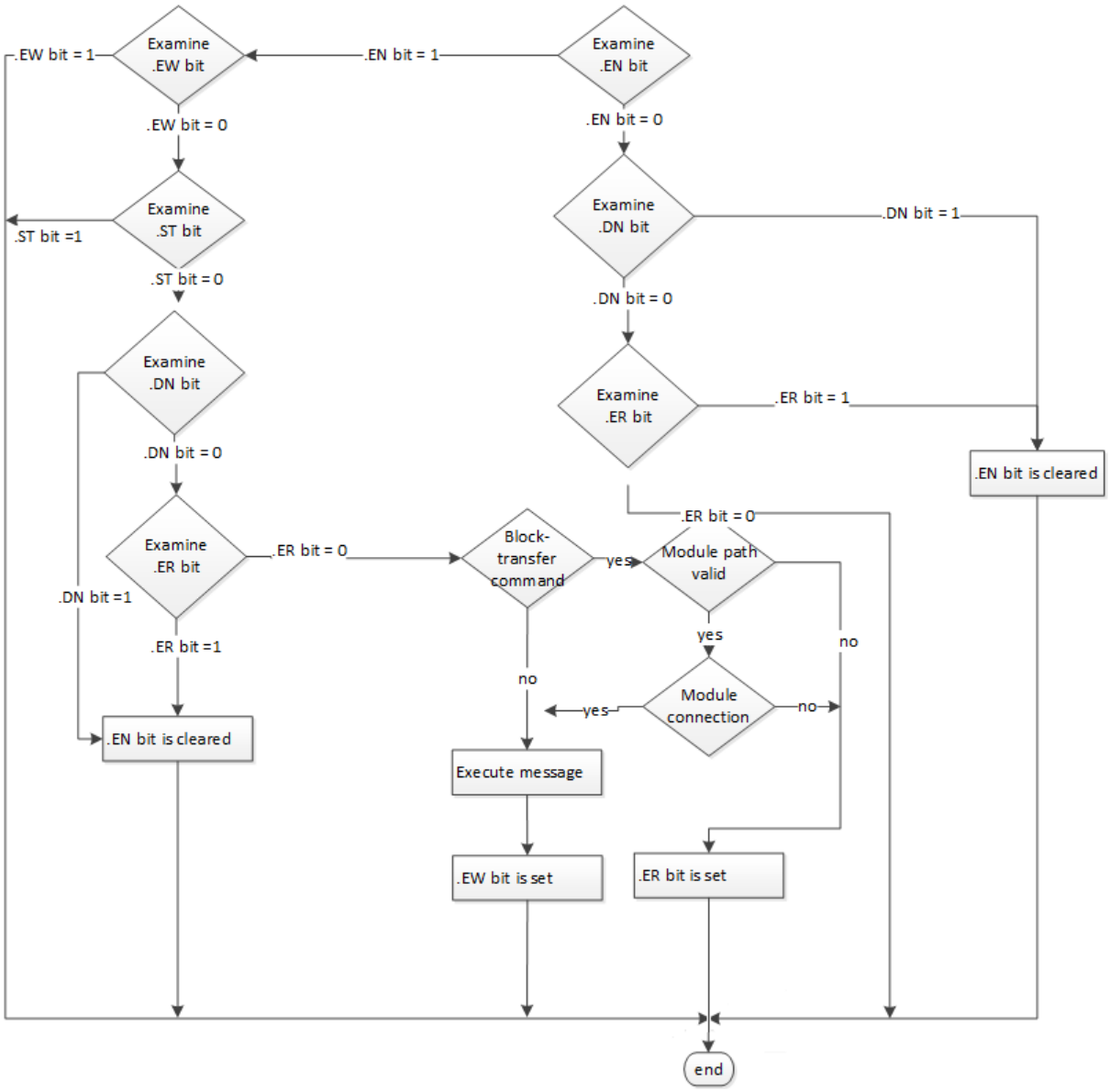
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	.EWS, .ST, .DN, .ER 비트가 해제됩니다.
렁-입력-조건이 거짓임	MSG 흐름도(거짓) 참조하십시오.
렁-입력-조건이 참임	MSG 흐름도(참) 참조하십시오.
사후 스캔	N/A

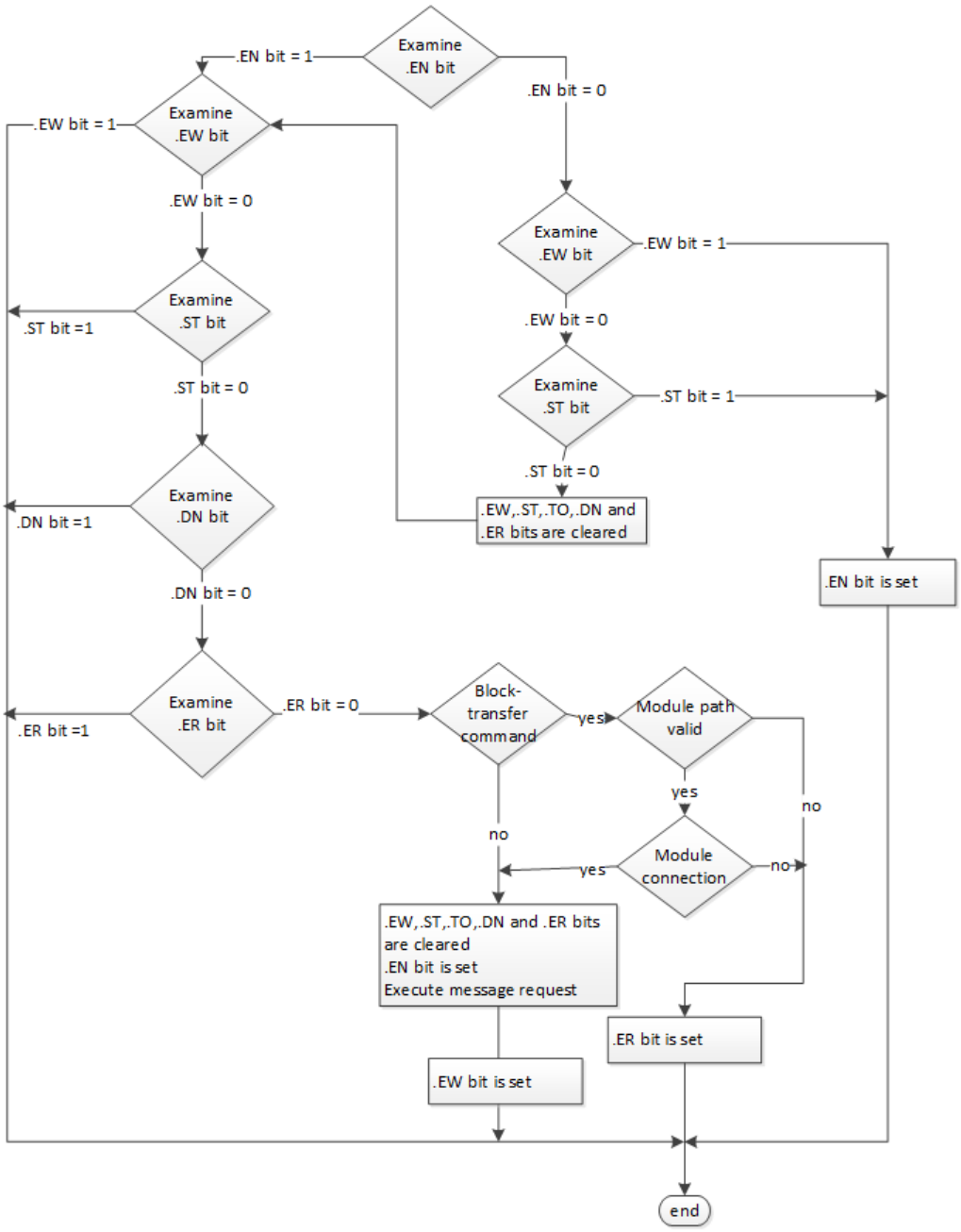
ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔 참조하십시오.
정상 실행	MSG 흐름도(참) 참조하십시오.
사후 스캔	래더 다이어그램 표의 사후 스캔 참조하십시오.

MSG 흐름도(거짓)


Examine .EW bit
.EW bit = 1
.EN bit = 1
Examine .EN bit
.EW bit = 0
.EN bit = 0
Examine .ST bit
.ST bit =1
Examine .DN bit
.DN bit = 1
.ST bit = 0
.DN bit = 0
Examine .DN bit
Examine .ER bit
.ER bit = 1
.EN bit is cleared
.DN bit = 0
.ER bit = 0
Examine .ER bit
.ER bit = 0
Block-transfer command
yes
Module path valid
.DN bit =1
.ER bit =1
yes
no
no
Module connection
yes
no
.EN bit is cleared
Execute message
.EW bit is set
.ER bit is set
end

MSG 흐름도(참)



예

래더 다이어그램

ST(스트럭쳐드 텍스트)

MSG(MessageControl);

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

메세지 에러 코드 페이지의 190

메세지 유형 선택 페이지의 298

통신 세부 정보 지정 페이지의 197

MSG 구성 예 페이지의 177

공통 속성 페이지의 963

MSG 구성 예

다음 예에 소스 및 대상 태그와 다양한 컨트롤러 조합의 요소가 나와 있습니다.

표에는 LOGIX 5000 컨트롤러에서 유래하여 다른 컨트롤러에 쓰여지는 MSG 명령어의 경로 설명이 나와 있습니다.

메세지 경로	소스 및 대상 예	
LOGIX 5000 -> LOGIX 5000	소스 태그	array_1[0]
	대상 태그	array_2[0]
	출처인 LOGIX 5000 컨트롤러의 소스 태그로 앨리어스 태그를 사용할 수 있습니다. 대상 태그로 앨리어스 태그를 사용할 수 없습니다. 대상 태그는 기본 태그여야 합니다.	
LOGIX 5000 -> PLC-5 LOGIX 5000 -> SLC	소스 태그	array_1[0]
	대상 요소	N7:10
	출처인 LOGIX 5000 컨트롤러에서 소스 태그로 앨리어스 태그를 사용할 수 있습니다.	
LOGIX 5000 -> PLC-2	소스 태그	array_1[0]
	대상 요소	010

아래 표에는 LOGIX 5000 컨트롤러에서 유래하거나 다른 컨트롤러에서 읽는 MSG 명령어의 경로에 대한 설명이 나와 있습니다.

메세지 경로	소스 및 대상 예	
LOGIX 5000 -> LOGIX 5000	소스 태그	array_1[0]
	대상 태그	array_2[0]
	앨리어스 태그를 소스 태그로 사용할 수 없습니다. 소스 태그는 기본 태그여야 합니다. 출처인 LOGIX 5000 컨트롤러에서는 앨리어스 태그를 대상 태그로 사용할 수 있습니다.	
LOGIX 5000 -> PLC-5 LOGIX 5000 -> SLC	소스 요소	N7:10
	대상 태그	array_1[0]
	출처인 LOGIX 5000 컨트롤러에서는 앨리어스 태그를 대상 태그로 사용할 수 있습니다.	
LOGIX 5000 -> PLC-2	소스 요소	010
	대상 태그	array_1[0]

추가 참조

메세지(MSG) 페이지의 166

메이저 폴트 유형 및 코드

메이저 폴트 목록에는 다음이 포함됩니다.

유형	코드	원인	복구 방법
1	1	컨트롤러가 실행 모드에서 전원이 켜졌습니다.	전원 켜기 핸들러를 실행하십시오.

1	16	I/O 통신 구성 폴트가 감지되었습니다.
		(CompactLogix 1768-L4x 컨트롤러만 해당됩니다.)
			컨트롤러 1768 버스 측에서 통신 모듈 수를 재구성하십시오.
			• 1768-L43 에 최대 2 개의 모듈이 있습니다.
			• 1768-L45 에 최대 4 개의 모듈이 있습니다.
			  • 최대 4개의 Sercos 모듈
			  • 최대 2개의 NetLinx 통신 모듈

1	40	컨트롤러가 배터리를 사용할 경우 전원 공급이 중단되면 배터리에 사용자 프로그램을 저장할 수 있는 충분한 전력이 없습니다.
		
		컨트롤러가 ESM(에너지 저장 모듈)을 사용할 경우 전원 공급이 중단되면 ESM 에 사용자 프로그램을 저장할 수 있는 충분한 전력이 없습니다.
			• 배터리를 사용하는 컨트롤러의 경우 배터리를 교체하십시오.
			• ESM(에너지 저장 모듈)을 사용하는 컨트롤러의 경우:
			  • 컨트롤러의 전원 공급이 중단되기 전에 ESM을 완충하십시오.
			  • ESM을 분리할 수 있는 경우 ESM을 교체하고, ESM을 분리할 수 없는 경우에는 컨트롤러를 교체하십시오.
			• 문제가 지속되면 Rockwell Automation 지원에 문의하십시오.

1	60	메모리 카드가 설치되지 않은 컨트롤러의 경우 컨트롤러가 다음을 수행했습니다. • 복구할 수 없는 폴트를 감지했습니다. • 메모리에서 프로젝트를 지웠습니다.	1. 폴트를 해결합니다. 2. 프로젝트를 다운로드합니다. 3. 원격 실행 또는 실행 모드로 변경합니다. 폴트가 지속될 경우: 1. 컨트롤러의 전원을 껐다가 다시 켜기 전에 OK 및 RS232 상태 표시기의 상태를 기록하십시오. 2. Rockwell Automation 지원에 문의하십시오.
1	61	메모리 카드가 설치된 컨트롤러의 경우 컨트롤러가 다음을 수행했습니다. • 복구할 수 없는 폴트를 감지했습니다. • 진단 정보를 메모리 카드에 기록했습니다. • 메모리에서 프로젝트를 지웠습니다.	1. 폴트를 해결합니다. 2. 프로젝트를 다운로드합니다. 3. 원격 실행/실행 모드로 변경합니다. 폴트가 지속되면 Rockwell Automation 지원에 문의하십시오.
1	62	보안 디지털(SD) 카드가 설치된 컨트롤러의 경우 컨트롤러가 다음을 수행했습니다. • 복구할 수 없는 폴트를 감지했습니다. • 진단 정보를 메모리 카드에 기록했습니다. 이 상태에서는 컨트롤러가 연결을 열지 않거나 실행 모드로의 전환을 허용하지 않습니다.	1. 폴트를 해결합니다. 2. 프로젝트를 다운로드합니다. 3. 원격 실행 또는 실행 모드로 변경합니다. 폴트가 지속되면 Rockwell Automation 지원에 문의하십시오.
3	16	필수 I/O 모듈 연결이 실패했습니다.	다음을 확인하십시오. • I/O 모듈이 섀시에 있습니다. • 전자 키 지정 요구 사항. • 폴트에 대한 자세한 내용은 컨트롤러 속성 메이저 폴트 탭과 모듈 속성 연결 탭을 참조하십시오.
3	20 21	섀시에 문제가 있을 수 있습니다.	복구할 수 없으므로 섀시를 교체하십시오.

3	23	실행 모드로 전환하기 전에 하나 이상의 필수 연결이 구성되었습니다.	실행 모드로 변경하기 전에 컨트롤러 I/O 표시등이 녹색으로 켜질 때까지 기다리십시오.
4	16	알 수 없는 명령어가 발생했습니다.	알 수 없는 명령어를 제거하십시오. 이는 프로그램 변환 프로세스로 인해 발생했을 수 있습니다.
4	20	배열 첨자가 너무 크고, 제어 구조 .POS 또는 .LEN 이 유효하지 않습니다.	정의된 범위 내로 값을 조정하십시오. 배열 크기 또는 정의된 차원을 초과하지 마십시오.
4	21	제어 구조 .LEN 또는 .POS < 0.	0 보다 커지도록 값을 조정하십시오.
4	31	JSR 명령어의 파라미터가 연결된 SBR 또는 RET 명령어의 파라미터와 일치하지 않습니다.	적절한 수의 파라미터를 전달하십시오. 너무 많은 파라미터를 전달하면 에러 없이 추가 파라미터가 무시됩니다.
4	34	타이머 명령어가 음수 사전 설정 값이거나 누산 값입니다.	음수 값을 타이머 사전 설정 또는 누산 값으로 로드하지 않도록 프로그램을 수정하십시오.
4	42	존재하지 않거나 삭제된 레이블에 대한 JMP 입니다.	JMP 대상을 수정하거나 누락된 레이블을 추가하십시오.
4	82	순차 펑션 차트(SFC)가 서브루틴을 호출하고 서브루틴이 호출 SFC 로 돌아가려고 했습니다. SFC 가 JSR 또는 FOR 명령어를 사용하여 서브루틴을 호출할 경우에 발생합니다.	호출 SFC 로 돌아가기를 제거하십시오.
4	83	테스트한 값이 필수 제한 범위 내에 없습니다. 이 에러는 부울 배열 및 비트 레벨 주소 지정에 사용된 배열 첨자에 발생합니다.	유효한 범위 내로 값을 조정하십시오. 배열 크기 또는 정의된 차원을 초과하지 마십시오.
4	84	스택 오버플로.	서브루틴 중첩 레벨 또는 전달되는 파라미터 수를 줄이십시오.
4	89	SFR 명령어에서 대상 루틴에 목표 단계가 없습니다.	SFR 대상을 수정하거나 누락된 단계를 추가하십시오.
4	90	안전 태스크 외부에서 안전 명령어를 사용하십시오.	안전 명령어를 안전 태스크 안에 포함하십시오.
4	91	장비 단계 명령어는 장비 단계 프로그램의 외부에서 호출합니다.	이 명령어는 장비 단계 프로그램에서만 사용하십시오.
4	94	중첩 제한이 초과되었습니다.	서브루틴 중첩 레벨을 줄이려면 프로젝트를 재구성하십시오.
4	990 - 999	사용자 정의 메이저 폴트.	

6	1	태스크 워치독이 만료되었습니다. 사용자 태스크가 지정된 기간에 완료되지 않았습니다. 프로그램 에러로 인해 무한 루프가 발생했거나, 프로그램이 너무 복잡해 지정한 대로 빠르게 실행할 수 없거나, 우선순위가 더 높은 태스크가 있어 이 태스크를 완료할 수 없습니다(단일 컨트롤러로 너무 많은 태스크를 수행하려고 함).	태스크 워치독을 높이거나, 실행 시간을 줄이거나, 이 태스크의 우선순위를 높이거나, 우선순위가 더 높은 태스크를 간소화하거나, 일부 코드를 다른 컨트롤러로 이동하십시오.
7	40	비휘발성 메모리에 저장하지 못했습니다.	• 프로젝트를 비휘발성 메모리에 다시 저장하십시오. • 프로젝트가 비휘발성 메모리에 저장되지 않을 경우 메모리 보드를 교체하십시오. • 1756-L7x 컨트롤러를 사용할 경우 SD 카드가 잠금 해제되었는지 확인하십시오.
7	41	컨트롤러 유형 불일치로 비휘발성 메모리를 로드하지 못했습니다.	올바른 유형의 컨트롤러로 변경하거나 프로젝트를 다운로드해서 메모리 카드에 저장하십시오.
7	42	비휘발성 메모리의 프로젝트 펌웨어 수정 버전이 컨트롤러의 펌웨어 수정 버전과 일치하지 않아 비휘발성 메모리를 로드하지 못했습니다.	컨트롤러 펌웨어를 비휘발성 메모리에 있는 프로젝트와 동일한 수정 버전 레벨로 업데이트하십시오.
7	43	잘못된 체크섬으로 인해 비휘발성 메모리를 로드하지 못했습니다.	Rockwell Automation 지원에 문의하십시오.
7	44	프로세서 메모리를 복원하지 못했습니다.	Rockwell Automation 지원에 문의하십시오.

7	50	로그 파일 인증서를 확인할 수 없습니다. 컨트롤러가 시작될 때 로그 파일 키/인증서 조합을 확인하려고 합니다. 확인에 따라 컨트롤러가 다음 작업 중 하나를 수행합니다.
		• 컨트롤러가 기존 로그 파일 인증서를 확인할 경우 기존 로그 디렉터리를 사용하여 계속합니다.
		• 기존 인증서를 확인할 수 없는 경우 컨트롤러가 메이저 폴트를 기록하고 새 인증서를 만들려고 합니다.
		  • 컨트롤러가 새 인증서를 성공적으로 만들면 백업 로그 하위 디렉터리를 만들고 기존 파일을 해당 디렉터리에 이동한 다음 새 확인 키 및 로그 파일 인증서를 사용하여 로깅 및 서명을 계속합니다.
		  • 컨트롤러가 새 인증서를 만들 수 없는 경우 로그 항목을 기존 로그 디렉터리에 기록하지만 해당 디렉터리의 서명 파일을 업데이트하지 않습니다.
			폴트를 해제하고 컨트롤러의 전원을 껐다가 켜십시오. 문제가 지속되면 Rockwell Automation 지원에 문의하십시오.

8	1	다운로드 중에 키 스위치를 사용하여 컨트롤러를 실행 모드로 전환하려고 했습니다.	다운로드가 완료될 때까지 기다린 후 폴트를 해제하십시오.
11	1	실제 위치가 양의 오버트래블 제한을 초과했습니다.	위치가 오버트래블 제한 범위 내에 올 때까지 축을 음의 방향으로 이동한 다음 모션 축 폴트 리셋을 실행하십시오.
11	2	실제 위치가 음의 오버트래블 제한을 초과했습니다.	위치가 오버트래블 제한 범위 내에 올 때까지 축을 양의 방향으로 이동한 다음 모션 축 폴트 리셋을 실행하십시오.
11	3	실제 위치가 위치 에러 허용 범위를 초과했습니다.	위치를 허용 범위 내로 이동한 다음 모션 축 폴트 리셋을 실행하십시오.
11	4	인코더 채널 A, B 또는 Z 연결이 끊어졌습니다.	인코더 채널을 다시 연결한 다음 모션 축 폴트 리셋을 실행하십시오.
11	5	인코더 노이즈 이벤트가 감지되었거나 인코더 신호가 구적에 없습니다.	인코더 케이블 연결을 수정한 다음 모션 축 폴트 리셋을 실행하십시오.
11	6	드라이브 폴트 입력이 활성화되었습니다.	드라이브 폴트를 해제한 다음 모션 축 폴트 리셋을 실행하십시오.
11	7	동기식 연결이 실패했습니다.	먼저 모션 축 폴트 리셋을 실행하십시오. 효과가 없으면 서보 모듈을 분리했다가 다시 연결하십시오. 그래도 실패할 경우 서보 모듈을 교체하십시오.
11	8	서보 모듈이 심각한 하드웨어 폴트를 감지했습니다.	모듈을 교체하십시오.
11	9	비동기식 연결이 실패했습니다.	먼저 모션 축 폴트 리셋을 실행하십시오. 효과가 없으면 서보 모듈을 분리했다가 다시 연결하십시오. 그래도 실패할 경우 서보 모듈을 교체하십시오.
11	10	모터 폴트가 발생했습니다.	자세한 내용은 DriveFaults 축 태그를 참조하십시오.
11	11	모터 열 폴트가 발생했습니다.	자세한 내용은 DriveFaults 축 태그를 참조하십시오.
11	12	모터 열 폴트가 발생했습니다.	자세한 내용은 DriveFaults 축 태그를 참조하십시오.
11	13	SERCOS 링 폴트가 발생했습니다.	SERCOS 광섬유 링 네트워크와 네트워크에 있는 장치의 무결성을 확인하십시오.
11	14	드라이브 활성화 입력 폴트가 발생했습니다.	드라이브 활성화 입력을 다시 활성화하고 폴트를 해제하십시오.

11	15	드라이브 결상 폴트가 발생했습니다.	드라이브에 대한 전체 전원 연결을 복원하고 폴트를 해제하십시오.
11	16	드라이브 가드 폴트가 발생했습니다.	자세한 내용은 GuardFaults 축 태그를 참조하십시오.
11	32	모션 태스크가 중복됩니다.	그룹의 기본 업데이트율이 너무 높아 올바른 작동을 유지할 수 없습니다. 그룹 폴트 태그를 지우고 그룹의 업데이트율을 높인 다음 메이저 폴트를 해제하십시오.

12	32	부적합한 2 차 컨트롤러의 전원을 껐다가 켰으며 전원을 켰을 때 파트너 섀시 또는 컨트롤러를 찾지 못했습니다.	다음을 확인하십시오.
			• 파트너 섀시가 연결되어 있습니다.
			• 전원이 중복 섀시에 모두 적용됩니다.
			• 파트너 컨트롤러에서 다음 항목이 동일합니다.
			  • 카탈로그 번호.
			  • 슬롯 번호.
			  • 펌웨어 수정 버전.

12	33	절체 후에 새 1 차 섀시에서 비파트너 컨트롤러가 확인되었습니다.	다음 중 하나를 수행하십시오. • 비파트너 컨트롤러를 제거하고 절체의 원인을 해결합니다. • 2 차 섀시에 파트너 컨트롤러를 추가합니다. • 절체의 원인을 해결하고 시스템을 동기화합니다.
12	34	절체가 발생한 직후에 1 차 및 2 차 컨트롤러의 키 스위치 위치가 일치하지 않습니다. 기존 1 차 컨트롤러는 프로그램 모드이고 새 1 차 컨트롤러는 실행 모드입니다.	다음 중 하나를 수행하십시오. • 키 스위치를 실행에서 프로그램으로 변경한 다시 실행 모드로 변경하여 폴트를 해제합니다. • Logix Designer 응용 프로그램을 사용하여 컨트롤러를 온라인 상태로 전환합니다. 그런 다음 폴트를 해제하고 두 컨트롤러의 모드를 실행으로 변경합니다.
14	1	안전 태스크 워치독이 만료되었습니다. 사용자 태스크가 지정된 기간에 완료되지 않았습니다. 프로그램 에러로 인해 무한 루프가 발생했거나, 프로그램이 너무 복잡해 지정한 대로 빠르게 실행할 수 없거나, 우선순위가 더 높은 태스크가 있어 이 태스크를 완료할 수 없거나, 안전 파트너가 제거되었습니다.	폴트를 해결합니다. 안전 태스크 서명이 있는 경우 안전 메모리가 다시 초기화되고 안전 태스크가 실행됩니다. 안전 태스크 서명이 없으면 안전 태스크가 실행될 수 있도록 프로그램을 다시 다운로드해야 합니다. 제거된 경우 안전 파트너를 다시 삽입하십시오.

14	2	안전 태스크의 루틴에 에러가 있습니다.	사용자 프로그램 로직에서 루틴의 에러를 해결하십시오.
14	3	안전 파트너가 누락되었습니다.	호환 가능한 안전 파트너를 설치하십시오.
14	4	안전 파트너를 사용할 수 없습니다.	호환 가능한 안전 파트너를 설치하십시오.
14	5	안전 파트너 하드웨어가 호환되지 않습니다.	호환 가능한 안전 파트너를 설치하십시오.
14	6	안전 파트너 펌웨어가 호환되지 않습니다.	호환 가능한 안전 파트너를 설치하십시오.
14	7	안전 태스크를 작동할 수 없습니다. 이 폴트는 1 차 컨트롤러와 안전 파트너 간에 로직이 일치하지 않거나, 워치독 타임아웃이 발생했거나, 메모리가 손상된 경우 등 안전 로직이 유효하지 않은 경우에 발생합니다.	폴트를 해결합니다. 안전 태스크 서명이 있는 경우 안전 메모리가 안전 태스크 서명을 사용하여 다시 초기화되고 안전 태스크가 실행됩니다. 안전 태스크 서명이 없으면 안전 태스크가 실행될 수 있도록 프로그램을 다시 다운로드해야 합니다.
14	8	CST(조정 시스템 시간) 마스터를 찾을 수 없습니다.	폴트를 해결합니다. 장치를 CST 마스터로 구성하십시오.
14	9	안전 파트너에 복구할 수 없는 컨트롤러 폴트가 발생했습니다.	폴트를 해제하고 프로그램을 다운로드하십시오. 폴트가 지속되면 안전 파트너를 교체하십시오.
17	34	컨트롤러 내부 온도가 작동 제한을 초과했습니다.	모듈의 주변 온도를 줄이기 위한 조치가 필요합니다. 주변(흡입구) 온도에 대한 권장 제한을 따르고 섀시 주위에 필요한 간격을 띄우십시오.
17	37	컨트롤러가 내부 온도 폴트에서 복구되었습니다.	컨트롤러가 자동 종료에서 복구되면 생성됩니다. 종료는 모듈의 온도가 유지 폴트의 온도 임계값을 초과할 경우에 발생합니다. 온도가 적절한 레벨로 감소하면 컨트롤러 전압이 다시 활성화되고 유형 17, 코드 37 폴트가 생성됩니다.
18	1	CIP Motion 드라이브가 올바르게 초기화되지 않았습니다.	시정 조치를 결정하려면 초기화 폴트 속성을 참조하여 발생한 폴트의 유형에 대한 자세한 내용을 확인하십시오.
18	2	CIP Motion 드라이브가 올바르게 초기화되지 않았습니다. 제조업체 관련 초기화 폴트가 발생한 경우 이 폴트가 표시됩니다.	시정 조치를 결정하려면 CIP 초기화 폴트 - Mfg 속성을 참조하여 발생한 폴트에 대한 자세한 내용을 확인하십시오.
18	3	물리적 축 폴트 비트가 설정되어 물리적 축에 폴트가 있음을 나타냅니다.	시정 조치를 결정하려면 CIP 축 폴트 속성을 참조하여 발생한 폴트에 대한 자세한 내용을 확인하십시오.

18	4	물리적 축 폴트 비트가 설정되어 물리적 축에 폴트가 있음을 나타냅니다. 제조업체 관련 축 폴트가 발생한 경우 이 폴트가 표시됩니다.	시정 조치를 결정하려면 CIP 초기화 폴트 - Mfg 속성을 참조하여 발생한 폴트에 대한 자세한 내용을 확인하십시오.
18	5	모션 폴트가 발생했습니다.	시정 조치를 결정하려면 모션 폴트 속성과 모션 폴트 비트를 참조하여 발생한 폴트에 대한 자세한 내용을 확인하십시오.
18	6	CIP Motion 드라이브 폴트가 발생했습니다. 일반적으로 이 폴트는 모듈과 관련된 모든 축에 영향을 미치며 관련된 모든 축이 종료됩니다.	폴트를 해결하려면 폴트가 발생한 모션 모듈을 재구성하십시오.
18	7	모션 그룹 폴트가 발생했습니다. 일반적으로 이 폴트는 모션 그룹과 관련된 모든 축에 영향을 미칩니다.	폴트를 해결하려면 전체 모션 하위 시스템을 재구성하십시오.
18	8	CIP Motion 드라이브 구성 도중 폴트가 발생했습니다. 일반적으로 이 폴트는 CIP Motion 드라이브의 축 구성 속성을 업데이트하려는 시도가 실패한 후에 발생합니다.	시정 조치를 결정하려면 모션 또는 1756-ENxT 모듈과 관련된 속성 오류 코드 및 속성 오류 ID 속성의 구성 폴트를 참조하십시오.
18	9	APR(절대 위치 복구) 폴트가 발생했으며 축의 절대 위치를 복구할 수 없습니다.	시정 조치를 결정하려면 APR 폴트를 참조하여 폴트의 원인을 확인하십시오.
18	10	APR(절대 위치 복구) 폴트가 발생했으며 축의 절대 위치를 복구할 수 없습니다. 제조업체 관련 APR 폴트가 발생한 경우 이 폴트가 표시됩니다.	시정 조치를 결정하려면 APR 폴트 - Mfg 속성을 참조하여 폴트의 원인을 확인하십시오.
18	128	가드 모션 안전 기능과 관련된 폴트가 발생했습니다. 이 폴트는 가드 안전 기능이 있는 드라이브를 사용할 경우에만 적용됩니다.	시정 조치를 결정하려면 가드 모션 속성 및 가드 상태 비트를 참조하여 폴트의 원인을 확인하십시오.
20	1	실행 또는 테스트 모드로 전환할 때 필요한 사용권이 없거나 만료되었습니다.	컨트롤러에서 프로젝트에 필요한 모든 사용권을 포함한 CmCard 를 삽입하십시오.

키워드: faults:4, fault code:1, fault codes:1

마이너 폴트 유형 및 코드

다음은 마이너 폴트 유형 및 코드입니다.

마이너 폴트 목록에는 다음이 포함됩니다.

유형	코드	원인	복구 방법
1	15	• 1769 전원 공급 장치가 컨트롤러의 1768 CompactBus 에 직접 연결되고 잘못된 구성을 사용하고 있습니다. • 1768 전원 공급 장치가 컨트롤러에 전원을 공급하지 못했습니다.	• 1768 CompactBus 의 전원 공급 장치를 제거하고 시스템의 전원을 껐다가 켭니다. • 전원 공급 장치를 교체하십시오.
3	1	버스 꺼짐 조건. 컨트롤러와 I/O 모듈 간 연결이 끊어졌습니다.	버스 꺼짐 폴트의 원인을 파악하려면 이 단계를 완료하십시오. 1. 프로젝트의 로컬 확장 모듈의 수가 시스템에 물리적으로 설치된 모듈의 수와 일치합니다. 2. 모든 장착 베이스가 잠겨 있고 I/O 모듈은 장착 베이스에 단단히 설치되어 있습니다. 3. 1734 POINT I/O 모듈은 모두 오토보 속도를 사용하도록 구성되었습니다. 이 단계로도 폴트 조건이 해결되지 않는 경우 Rockwell Automation 지원에 문의하십시오.
3	94	I/O 모듈의 현재 RPI 업데이트가 이전 RPI 업데이트와 중복됩니다.	I/O 모듈의 RPI 속도를 더 큰 숫자 값으로 설정하십시오. Rockwell Automation 은 모듈 RPI 중복 폴트 상태에서는 CompactLogix 5370 L2 및 CompactLogix 5370 L3 제어 시스템을 실행하지 않도록 권장합니다.
3	100	입력/출력 크기 중 하나 또는 모두가 16 바이트를 초과하고 모듈이 시작 및 종료 무결성을 지원하지 않으므로 모듈에 데이터 무결성 손실 가능성이 존재합니다.	복구 방법: • 입력/출력 크기를 16 바이트 이하로 줄여 데이터 무결성 손실 문제를 방지하십시오. • 모듈 제공자에게 연락하여 시작 및 종료 무결성 기능을 지원하는 버전에 대해 문의하십시오. • 자세한 내용은 Rockwell Automation Knowledgebase 답변 ID 1028837을 참조하십시오.
4	4	명령에 산술 오버플로가 발생했습니다.	산술 연산자(순서)를 검사하거나 값을 조정하여 프로그램을 수정합니다.

4	5	GSV/SSV 명령어에서 지정된 인스턴스를 찾을 수 없습니다.	인스턴스 이름을 확인하십시오.
4	6	GSV/SSV 명령어에서 다음 중 하나가 발생했습니다. • 지정한 클래스 이름이 지원되지 않습니다. • 지정한 속성 이름이 잘못되었습니다.	클래스 이름과 속성 이름을 확인하십시오.
4	7	GSV/SSV 대상 태그는 모든 데이터를 보관하기에 너무 작습니다.	충분한 공간을 갖도록 대상 또는 소스를 수정하십시오.
4	30	잘못된 파라미터가 ASCII 포트로 통과했습니다.	ASCII 구성 설정을 확인하십시오.
4	35	PID 델타 시간이 ≤ 0 입니다.	PID 델타 시간을 0 보다 큰 값으로 조정하십시오.
4	36	PID 설정값이 범위를 벗어났습니다.	범위 내 값으로 설정값을 조정하십시오.
4	51	문자열 태그의 LEN 값이 문자열 태그의 DATA 크기보다 큽니다.	• 문자열 태그의 LEN 구성원에 쓰고 있는 명령어가 없는지 확인합니다. • LEN 값에 문자열에 포함되는 문자의 수를 입력합니다.
4	52	출력 문자열이 대상보다 큽니다.	출력 문자열에 대한 충분한 크기의 새 문자열 데이터 유형을 만듭니다. 새 문자열 데이터 유형을 대상의 데이터 유형으로 사용합니다.
4	53	출력 수가 대상 데이터 유형의 제한을 초과합니다.	다음 이유로 인해 발생합니다: • ASCII 값의 크기를 줄입니다. • 대상에 대해 더 큰 데이터 유형을 사용합니다.
4	56	Start 및 Quantity 값이 유효하지 않습니다.	• Start 값이 1 과 Source 의 DATA 크기 사이에 있는지 확인하십시오. • Start 값과 Quantity 값의 합이 Source 의 DATA 크기 이하인지 확인하십시오.
4	57	직렬 포트가 핸드셰이킹 없음으로 설정되었기 때문에 AHL 명령어가 실행되지 못합니다.	다음 이유로 인해 발생합니다: • 직렬 포트의 제어 라인 설정을 변경합니다. • AHL 명령어를 삭제합니다.
6	2	주기 태스크 오버랩 다시 실행할 때까지 주기 태스크가 완료되지 않습니다.	프로그램을 간단히 하거나 기간을 늘리거나 상대적 우선 순위를 높이는 등 변경 작업을 수행합니다.
6	3	이벤트 태스크가 중복됩니다. 다시 실행할 때까지 이벤트 태스크가 완료되지 않습니다.	프로그램을 간단히 하거나 기간을 늘리거나 상대적 우선 순위를 높이거나 트리거링 이벤트를 느리게 하는 등 변경 작업을 수행합니다.

7	49	컨트롤러는 비휘발성 메모리에서 프로젝트를 로드할 때 이 마이너 폴트를 기록하고 FaultLog 객체, MinorFaultBits 속성, 비트 7 을 설정합니다.	폴트를 해결합니다.
9	0	시리얼 포트를 사용하는 동안 알 수 없는 에러가 발생했습니다.	문제가 지속되면 Rockwell Automation 기술 지원에 문의하십시오.
9	1	현재 구성에 대해 CTS 라인이 올바르지 않습니다.	컨트롤러에 대한 시리얼 포트 케이블의 연결을 끊고 다시 연결합니다. 케이블 연결이 올바른지 확인하십시오.
9	2	폴링 목록 에러입니다. DF1 마스터의 폴링 목록에서 파일 크기보다 더 많은 스테이션 지정, 255 개보다 많은 스테이션 지정, 목록의 끝을 지나는 인덱싱 시도 또는 브로드캐스트 주소 폴링(STN #255)과 같은 폴트가 감지되었습니다.	다음 에러에 대해 확인하십시오. • 스테이션의 총 수가 폴링 목록 태그의 공간보다 큽니다. • 스테이션의 총 수가 255 보다 큽니다. • 현재 스테이션 포인터가 폴링 목록 태그의 끝보다 큽니다. • 254 보다 큰 스테이션 번호가 발견되었습니다.
9	3	RS-232 DF1 마스터 활성 스테이션 태그가 지정되지 않았습니다.	컨트롤러 속성의 시리얼 포트 프로토콜 탭에서 활성 스테이션 태그로 사용될 태그를 지정하십시오.
9	5	DF1 슬레이브 폴링 시간 초과입니다. 슬레이브에 대한 폴링 워치독의 시간이 초과되었습니다. 마스터가 지정된 시간 내에 이 컨트롤러를 폴링하지 않았습니다.	폴링 지연을 확인하고 수정합니다.
9	9	모뎀 연결이 끊어졌습니다. DCD 또는 DSR 제어 라인이 올바른 시퀀스 및/또는 상태로 수신되고 있지 않습니다.	컨트롤러에 대한 모뎀 연결을 수정합니다.
9	10	데이터가 시리얼 포트에서 제거되거나 손실되었습니다.	초기화 프로그램이 데이터를 보내는 속도를 늦춥니다.
10	10	배터리가 감지되지 않거나 교체되어야 합니다.	새 배터리를 설치하십시오.
10	11	안전 파트너 배터리가 감지되지 않거나 교체되어야 합니다.	새 배터리를 설치하십시오.
10	12	ESM(에너지 저장 모듈)이 설치되지 않았습니다. 컨트롤러의 전원이 꺼진 경우 WallClockTime 속성 및 프로그램이 유지되지 않습니다.	컨트롤러에 ESM 을 설치하십시오.
10	13	설치된 ESM 이 컨트롤러와 호환되지 않습니다.	설치된 ESM 을 컨트롤러와 호환되는 ESM 으로 교체하십시오.

10	14	ESM 을 하드웨어 폴트로 인해 교체해야 합니다. 전원이 꺼지면 WallClockTime 속성 또는 컨트롤러 프로그램을 유지할 수 없습니다.	ESM 을 교체하십시오.
10	15	전원이 꺼지면 ESM 이 WallClockTime 속성 또는 컨트롤러 프로그램을 유지하기에 충분한 에너지를 ESM 에 저장할 수 없습니다.	ESM 을 교체하십시오.
10	16	UPS(무정전 전원장치)가 없거나 준비되지 않았습니다.	다음 이유로 인해 발생합니다: • UPS 를 설치하십시오. • UPS 를 점검하여 정전 시에 백업 전원을 제공하기에 충분히 충전되어 있는지 확인하십시오.
10	17	UPS 배터리에 정애가 생겼고 교체가 필요합니다.	UPS 의 배터리를 교체하십시오.
13	21	벽시계 시간이 범위를 벗어났습니다.	벽시계 시간이 올바른 날짜/시간으로 설정되었는지 확인하십시오.
14	12	안전 프로젝트는 SIL2/PLd 로 구성되고 안전 파트너가 있습니다.	안전 파트너가 1 차 컨트롤러의 오른쪽에 설치되지 않았는지 확인하십시오.
17	1...*n*	내부 컨트롤러 진단이 실패했습니다.	폴트 유형 및 폴트 코드는 Rockwell Automation 기술 지원에 문의하십시오.
17	35	컨트롤러 내부 온도가 작동 제한에 근접했습니다.	모듈의 주변 온도를 줄이기 위한 조치가 필요합니다. 주변(흡입구) 온도에 대한 권장 제한을 따르고 섀시 주위에 필요한 간격을 띄우십시오.
17	36	팬이 없거나 원하는 속도를 유지하지 않습니다.	팬을 교체하십시오.
19	4	Ethernet 포트 폴트	EtherNet/IP 데이터 스톰이 감지되었습니다. Ethernet 포트의 네트워크 트래픽을 확인하고 폴트를 해제하십시오. 문제가 지속되면 Rockwell Automation 기술 지원에 추가 지원을 문의하십시오.
20	1	컨트롤러가 실행 또는 테스트 모드일 때 필요한 사용권이 없거나 만료되었습니다.	컨트롤러에서 프로젝트에 필요한 모든 사용권을 포함한 CmCard 를 삽입하십시오.

키워드: 폴트 코드: 2, 폴트 코드: 2, 폴트: 2 키워드: 폴트:5

메세지 에러 코드

에러 코드는 MSG 명령어 유형에 따라 다릅니다.

추가 참조

에러 코드 페이지의 191

확장 에러 코드 페이지의 192

PLC 및 SLC 에러 코드(.ERR) 페이지의 194

블록 전송 에러 코드 페이지의 196

에러 코드

Logix Designer 응용 프로그램이 전체 설명을 표시하지 않는 경우도 있습니다.

에러 코드(16 진수)	설명	소프트웨어에 표시되는 내용
0001	연결 실패(확장된 에러 코드)	설명과 동일
0002	리소스 부족	
0003	유효하지 않은 값	
0004	IOI 구문 에러(확장 에러 코드 참조)	
0005	대상을 알 수 없음, 클래스 지원되지 않음, 인스턴스 정의되지 않음 또는 구조 요소 정의되지 않음(확장 에러 코드 참조)	
0006	패킷 공간 부족	
0007	연결 끊어졌음	
0008	서비스 지원되지 않음	
0009	데이터 세그먼트 에러 또는 잘못된 속성 값	
000A	속성 목록 에러	
000B	상태 이미 존재함	
000C	객체 모델 충돌	
000D	객체 이미 존재함	
000E	속성 값 설정될 수 없음	
000F	권한 거부됨	
0010	장치 상태 충돌	
0011	응답이 맞지 않음	
0012	원시 조각화	
0013	명령 데이터 부족	
0014	속성 지원되지 않음	
0015	데이터 너무 많음	
001A	브리지 요청 너무 큼	
001B	브리지 응답 너무 큼	
001C	속성 목록 부족	
001D	잘못된 속성 목록	설명과 동일
001E	포함된 서비스 에러	
001F	연결 관련 실패(확장 에러 코드 참조)	

0022	잘못된 응답 수신됨	
0025	키 세그먼트 에러	
0026	잘못된 IOI 에러	
0027	목록에 포함된 예기치 않은 속성	
0028	DeviceNet 에러 - 잘못된 구성원 ID	
0029	DeviceNet 에러 - 구성원 설정할 수 없음	
00D1	모듈이 실행 상태 아님	알 수 없는 에러
00FB	메세지 포트 지원되지 않음	
00FC	메세지 데이터 유형 지원되지 않음	
00FD	메세지가 초기화되지 않음	
00FE	메세지 타임아웃	
00FF	일반 에러(확장 에러 코드 참조)	

확장 에러 코드

Logix Designer 응용 프로그램이 확장 에러 코드에 대해 어떠한 텍스트도 표시하지 않습니다.

다음은 에러 코드 0001 에 대한 확장 에러 코드입니다.

확장 에러 코드(16 진수)	설명
0100	사용 중인 연결
0103	전송 지원되지 않음
0106	소유권 충돌
0107	연결 찾지 못함
0108	잘못된 연결 유형
0109	잘못된 연결 크기
0110	모듈 구성되지 않음
0111	EPR 지원되지 않음
0113	MSG 쓰기 실패
0114	잘못된 모듈
0115	잘못된 장치 유형
0116	잘못된 버전
0118	유효하지 않은 구성 형식

011A	응용 프로그램 연결에서 벗어남
0203	연결 타임아웃
0204	연결되지 않은 메세지 타임아웃
0205	연결되지 않은 보내기 파라미터 에러
0206	메세지 너무 큼
0301	버퍼 메모리 없음
0302	대역폭 사용 불가
0303	사용 가능한 화면 없음
0305	서명 일치하지 않음
0311	포트 사용 불가
0312	링크 주소 사용 불가
0315	유효하지 않은 세그먼트 유형
0317	연결 예약되지 않음

다음은 에러 코드 001F 에 대한 확장 에러 코드입니다.

확장 에러 코드(16 진수)	설명
0203	연결 타임아웃

다음은 에러 코드 0004 및 0005 에 대한 확장 에러 코드입니다.

확장 에러 코드(16 진수)	설명
0000	확장 상태가 메모리를 벗어남
0001	확장 상태가 인스턴스를 벗어남

다음은 에러 코드 00FF 에 대한 확장 에러 코드입니다.

확장 에러 코드(16 진수)	설명
2001	과도한 IOI
2002	잘못된 파라미터 값
2018	세마포 거부
201B	크기 너무 작음
201C	유효하지 않은 크기
2100	권한 실패

2101	유효하지 않은 키 스위치 위치
2102	유효하지 않은 암호
2103	암호 발행되지 않음
2104	주소가 범위를 벗어남
2105	주소 및 개수가 범위를 벗어남
2106	사용 중인 데이터
2107	유형이 유효하지 않거나 지원되지 않음
2108	컨트롤러가 업로드 또는 다운로드 모드임
2109	배열 차원 수를 변경하려고 시도
210A	유효하지 않은 기호 이름
210B	기호 없음
210E	검색 실패
210F	태스크를 시작할 수 없음
2110	쓸 수 없음
2111	읽을 수 없음
2112	공유 루틴은 편집할 수 없음
2113	컨트롤러가 폴트 모드임
2114	실행 모드 금지됨

PLC 및 SLC 에러 코드(.ERR)

Logix 펌웨어 버전 10.x 이상에서는 PLC 및 SLC™ 메세지 유형(PCCC 메세지)과 관련된 에러에 대해 새 에러 코드를 제공합니다.

이러한 변경으로 인해 RSLogix 5000 소프트웨어에는 많은 에러에 대한 보다 더 의미 있는 설명이 표시됩니다. 이전에 소프트웨어에는 00F0 에러 코드와 관련된 에러에 대한 설명이 표시되지 않았습니다.

또한 이번 변경으로 인해 코드 에러가 다른 컨트롤러(예: PLC-5® 컨트롤러)에서 반환하는 에러와의 일관성이 더욱 향상됩니다.

다음 표에는 R9.x 이전 버전부터 R10.x 이상 버전까지 에러 코드에 대한 변경 사항이 나와 있습니다. 이번 변경의 결과, .ERR

구성원은 각 PCCC 에러에 대해 고유한 값을 반환합니다. 이러한 에러에는 EXERR 이 더 이상 필요하지 않습니다.

PLC 및 SLC 에러 코드(16 진수)

R9.x 이전 버전		R10.x 이상 버전		설명
.ERR	.EXERR	.ERR	.EXERR	
0010		1000		잘못된 로컬 프로세서의 명령 또는 형식
0020		2000		통신 모듈이 작동하지 않음
0030		3000		원격 노드가 없거나, 연결이 끊겼거나, 종료됨
0040		4000		프로세서가 연결되었으나 폴트가 발생함(하드웨어)
0050		5000		잘못된 스테이션 번호
0060		6000		요청된 펑션 사용 불가
0070		7000		프로세서가 프로그램 모드임
0080		8000		프로세서의 호환성 파일이 없음
0090		9000		원격 노드가 명령을 버퍼링할 수 없음
00B0		B000		프로세서가 다운로드 중이므로 프로세서에 액세스할 수 없음
00F0	0001	F001		프로세서가 주소를 잘못 변환함
00F0	0002	F002		불완전한 주소
00F0	0003	F003		잘못된 주소
00F0	0004	F004		잘못된 주소 형식 - 기호 없음
00F0	0005	F005		잘못된 주소 형식 - 기호의 크기가 장치에서 지원하는 최대 문자 수보다 크거나 0 임
00F0	0006	F006		목표 프로세서에 주소 파일이 없음
00F0	0007	F007		요청된 워드 수에 비해 대상 파일이 너무 작음
00F0	0008	F008		요청을 완료할 수 없음 다중 패킷 작업 중 상황이 변경됨
00F0	0009	F009		데이터 또는 파일이 너무 큼 메모리 사용 불가
00F0	000A	F00A		목표 프로세서가 요청된 정보를 패킷에 저장할 수 없음
00F0	000B	F00B		권한 에러, 액세스 거부됨
00F0	000C	F00C		요청된 펑션 사용 불가

00F0	000D	F00D		중복 요청
00F0	000E	F00E		명령 실행할 수 없음
00F0	000F	F00F		오버플로, 히스토그램 오버플로
00F0	0010	F010		액세스 불가
00F0	0011	F011		요청된 데이터 유형이 사용 가능한 데이터와 일치하지 않음
00F0	0012	F012		잘못된 명령 파라미터
00F0	0013	F013		주소 참조가 삭제된 영역에 있음
00F0	0014	F014		알 수 없는 이유로 명령 실행 실패 PLC-3® 히스토그램 오버플로
00F0	0015	F015		데이터 변환 에러
00F0	0016	F016		1771 랙 어댑터와 통신하는 데 스캐너 사용 불가
00F0	0017	F017		모듈과 통신하는 데 어댑터 사용 불가
00F0	0018	F018		1771 모듈 응답이 유효하지 않음
00F0	0019	F019		라벨 중복
00F0	001A	F01A		파일 소유자 활성 상태 - 파일이 사용 중임
00F0	001B	F01B		프로그램 소유자 활성 상태 - 온라인으로 다운로드하거나 편집하는 사람이 있음
00F0	001C	F01C		디스크 파일이 쓰기 보호되어 있거나 디스크 파일에 액세스할 수 없음(오프라인 전용).
00F0	001D	F01D		다른 응용 프로그램에서 디스크 파일 사용 중임 업데이트가 수행되지 않음(오프라인 전용).

블록 전송 에러 코드

다음은 LOGIX 5000 블록 전송 관련 에러 코드입니다.

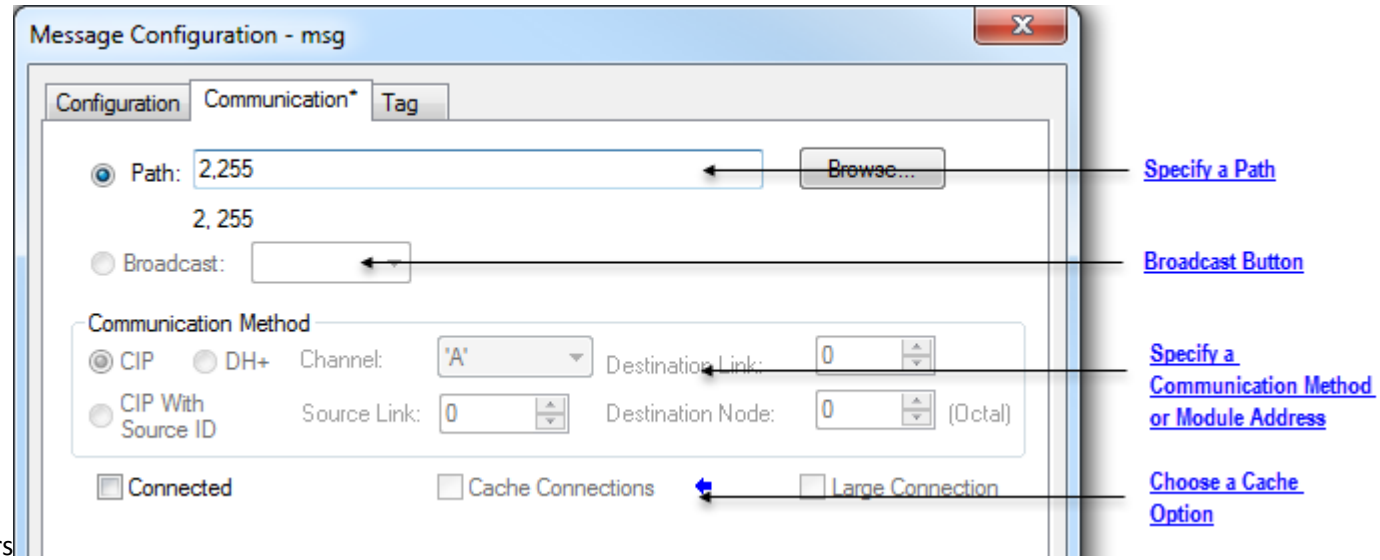
에러 코드(16 진수)	설명	소프트웨어에 표시되는 내용
00D0	스캐너가 요청 후 3.5 초 이내에 블록 전송 모듈에서 블록 전송 응답을 받지 못했습니다.	알 수 없는 에러
00D1	읽기 응답의 체크섬이 데이터 스트림의 체크섬과 일치하지 않았습니다.	
00D2	스캐너가 읽기 또는 쓰기를 요청했으나 블록 전송 모듈이 반대로 응답했습니다.	
00D3	스캐너가 길이를 요청했으나 블록 전송 모듈이 다른 길이로 응답했습니다.	

00D6	스캐너가 블록 전송 모듈로부터 쓰기 요청에 실패했음을 나타내는 응답을 수신했습니다.	
00EA	스캐너가 이 블록 전송 모듈을 포함하고 있을 수 있는 랙과 통신하도록 구성되지 않았습니다.	
00EB	지정된 논리 슬롯을 지정된 랙 크기에 사용할 수 없습니다.	
00EC	현재, 처리 중인 블록 전송 요청이 있어 다른 요청을 시작하려면 응답이 필요합니다.	
00ED	블록 전송 요청의 크기가 유효한 블록 전송 크기 요청과 일치하지 않습니다.	
00EE	블록 전송 요청 유형이 예상 BT_READ 또는 BT_WRITE 와 일치하지 않습니다.	
00EF	스캐너가 블록 전송 요청을 수용하기 위해 블록 전송 표에서 사용 가능한 슬롯을 찾을 수 없습니다.	
00F0	해결되지 않은 블록 전송이 있는 상태에서 스캐너가 원격 I/O 채널을 리셋하라는 요청을 받았습니다.	
00F3	원격 블록 전송 큐가 꽉 찼습니다.	
00F5	요청된 랙 또는 슬롯에 대해 구성된 통신 채널이 없습니다.	
00F6	원격 I/O 에 대해 구성된 통신 채널이 없습니다.	
00F7	명령어에 설정된 블록 전송 타임아웃이 완료 전 타임아웃됩니다.	
00F8	블록 전송 프로토콜 에러 - 청하지 않는 블록 전송.	
00F9	잘못된 통신 채널로 인해 블록 전송 데이터가 손실됩니다.	
00FA	블록 전송 모듈이 관련된 블록 전송 명령어와 다른 길이를 요청했습니다.	
00FB	블록 전송 읽기 데이터의 체크섬이 잘못되었습니다.	
00FC	어댑터와 블록 전송 모듈 간에 블록 전송 쓰기 데이터 전송이 잘못되었습니다.	
00FD	블록 전송 크기와 블록 전송 데이터 표의 인덱스 크기를 더한 값이 블록 전송 데이터 표 파일의 크기보다 컸습니다.	

통신 세부 정보 지정

래더 로직 또는 ST(스트럭처드 텍스트) 프로그램으로 브로드캐스트를 설정합니다. 래더 로직의 경우 렁을 추가하고 **MSG** 속성을 클릭하여 **메세지 구성**(Message Configuration) 대화 상자에 액세스하고 새 메세지를 설정합니다. ST(스트럭처드 텍스트)에서는 **MSG**(aMsg)를 입력한 다음 aMsg 를 마우스 오른쪽 단추로 클릭하여 **메세지 구성**(Message Configuration) 대화 상자에 액세스하고 메세지를 구성합니다.

MSG 명령어를 구성하려면 **통신**(Communication) 탭에서 다음 항목을 지정합니다.



rs

경로 지정

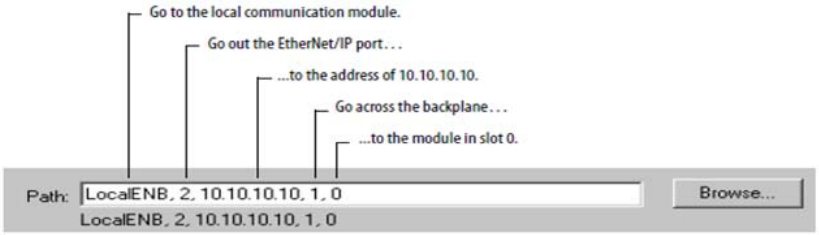
경로에는 메세지가 대상에 도달하기 위해 선택하는 경로가 표시됩니다. 컨트롤러의 I/O 구성 이름 또는 입력한 숫자를 사용하거나 두 가지를 모두 사용합니다. 브로드캐스트 단추를 사용하여 경로를 기본값으로 설정할 수 있으며 시스템 프로토콜 및 메세지 유형을 포함하여 활성화되어야 합니다.

다음의 경우	그러면
컨트롤러의 I/O 구성에 메세지 수신 모듈이 있는 경우	"찾아보기"(Browse)를 사용하여 모듈을 선택합니다.
컨트롤러의 I/O 구성에 로컬 통신 모듈만 있는 경우	"찾아보기"(Browse)를 사용하여 로컬 통신 모듈을 선택하고 나머지 경로를 입력합니다.
컨트롤러의 I/O 구성에 메세지에 필요한 모듈이 없는 경우	경로를 입력합니다.

팁: THIS 또한 사용할 수 있으며 컨트롤러 자체 경로를 나타냅니다. 컨트롤러에 연결되지 않은 메세지를 보낼 때 THIS 를 사용합니다.

예제

컨트롤러의 I/O 구성에 로컬 통신 모듈만 있는 경우:



경로를 입력하려면 다음 형식을 사용합니다.

port, next_address, port, next_address,

위치	상태	
	사용 네트워크	유형
포트	백플레인	1
	DF1(시리얼, 시리얼 채널 0)	2
	ControlNet	
	EtherNet/IP	
	DH+ 채널 A	
	DH+ 채널 B	3
	DF1 채널 1(시리얼 채널 1)	
Next_address	백플레인	모듈의 슬롯 번호
	DF1(시리얼)	스테이션 주소(0-254)
	ControlNet	노드 번호(1-99 10 진수)
	DH+	8# 다음에 노드 번호(1-77 8 진수) 예를 들어 8 진수 노드 주소인 37 을 지정하려면 8#37 을 입력합니다.
	EtherNet/IP	다음 형식 중 하나를 사용하여 EtherNet/IP 네트워크의 모듈을 지정합니다. • IP 주소. 예: 10.10.10.10 • IP 주소:포트. 예: 10.10.10.10:24 • DNS 이름. 예: tanks • DNS 이름:포트. 예: tanks:24

브로드캐스트 단추

브로드캐스트(Broadcast) 단추는 시리얼 포트에서 사용됩니다.

- RSLogix 5000 소프트웨어의 경우 버전 18 부터 이 기능을 통해 메세지를 대상으로 전송하는 데 필요한 루트 및 메세지 유형의 정의 기능이 향상됩니다.

활성화된 **브로드캐스트**(Broadcast) 단추를 사용하면 콤보 상자에서 사용 가능한 채널을 선택하는 식으로 경로를 기본값으로 설정할 수 있습니다. 콤보 상자에 나열되는 채널의 수는 현재 컨트롤러에 따라 다릅니다.

기본적으로 **통신**(Communication) 탭의 **경로**(Path) 단추는 활성입니다.

브로드캐스트(Broadcast) 단추를 활성화하고 메세지 기본 경로를 설정하는 채널을 선택하려면 다음 단계를 수행합니다.

1. **컨트롤러 관리자**(Controller Organizer)에서 **컨트롤러**(Controller)를 마우스 오른쪽 단추로 클릭한 다음 **속성**(Properties)을 선택합니다. **컨트롤러 속성**(Controller Properties) 대화 상자가 나타납니다.
2. **시스템 프로토콜**(System Protocol) 탭을 클릭합니다.
3. **프로토콜**(Protocol) 상자에서 **DF1 마스터**(DF1 Master)를 선택합니다. 폴링 모드는 기본적으로 '메세지 기반'(Message Based)으로 됩니다(슬레이브가 메세지를 시작할 수 있음).
4. **확인**(OK)을 클릭합니다.
5. 래더 로직에서 MSG 태그 내부의 상자를 클릭합니다. **메세지 구성**(Message Configuration) 대화 상자에 **구성**(Configuration) 탭이 열린 채로 표시됩니다.
6. **메세지 유형**(Message Type) 상자에서 **CIP 데이터 표 쓰기**(CIP Data Table Write)를 선택합니다.
7. **확인**(OK)을 클릭합니다. **통신**(Communication) 탭의 **브로드캐스트**(Broadcast) 단추가 활성화됩니다.
8. **통신**(Communication) 탭을 클릭합니다.
9. **브로드캐스트**(Broadcast) 단추 옆 콤보 상자에서 채널을 선택합니다. 콤보 상자의 채널 수는 컨트롤러에 따라

다릅니다.
채널 0 또는 1 을 선택할 경우 **메세지 구성**(Message Configuration) 대화 상자의 해당 메세지 경로는 기본적으로 2,255(채널 0) 또는 3,255(채널 1)로 설정됩니다. Path(경로)가 회색으로 비활성화되어 직접 경로 값을 입력할 수 없습니다.

10. **확인**(OK)을 클릭합니다.

시스템 프로토콜 탭 구성

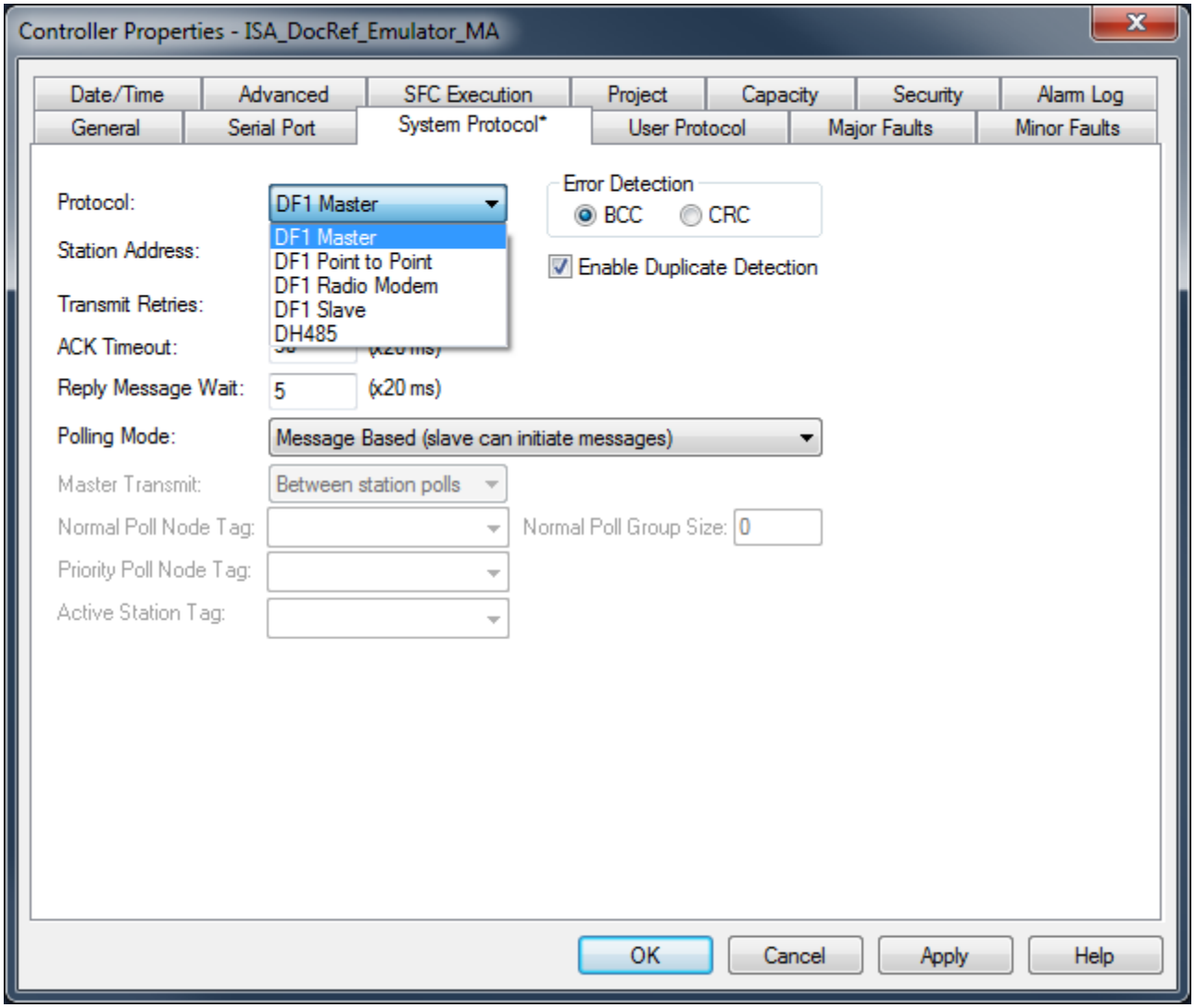
Logix Designer 응용 프로그램의 ControlLogix 컨트롤러에서 브로드캐스트를 실행하려면 **컨트롤러 속성**(Controller Properties) 대화 상자의 **시스템 프로토콜**(System Protocol) 탭을 구성해야 합니다. **메세지 구성**(Message Configuration) 대화 상자의 '쓰기' 메세지 유형과 프로토콜이 호환되어야 합니다.

아래 단계에 따라 브로드캐스트 기능과 호환되도록 시스템 프로토콜을 설정하십시오.

1. 응용 프로그램에서 컨트롤러를 생성하거나 기존 컨트롤러를 엽니다.

2. **컨트롤러 관리자**(Controller Organizer)에서 컨트롤러 이름을 마우스 오른쪽 단추로 클릭하고 **속성**(Properties)을 선택합니다. **컨트롤러 속성**(Controller Properties) 대화 상자가 표시됩니다.

3. 컨트롤러에 시리얼 포트가 있는 경우 **시스템 프로토콜**(System Protocol) 탭을 클릭합니다.



4. 프로토콜(Protocol) 상자에서 프로토콜을 선택합니다.

중요	**메세지 구성 탭**(Message Configuration Tab) 대화 상자의 **메세지 유형**(Message Type) 상자가 시스템 프로토콜과 호환되도록 쓰기 유형이라야 합니다. 그렇지 않은 경우 **브로드캐스트**(Broadcast) 단추가 비활성화됩니다.

5. 아래 표에 간략히 기술된 각 프로토콜에 대해 **시스템 프로토콜**(System Protocol) 탭의 정보를 입력합니다.

항목	설명
Protocol	DF-1 마스터(DF-1 Master)
스테이션 주소(Station Address)	컨트롤러 스테이션 주소 번호 입력
전송 다시 시도(Transmit Retries)	3
ACK 타임아웃(ACK Timeout)	50
응답 메세지 대기(Reply Message Wait)	5
폴링 모드(Polling Mode)	다음 모드 중 선택합니다. • **메세지 기반**(Message Based) 메세지 명령어를 사용하는 슬레이브 폴링 • **슬레이브가 메세지를 시작할 수 있음**(Slave can initiate message) 슬레이브 간 브로드캐스트용 • **표준**(Standard). 슬레이브 폴링 일정
에러 감지(Error Detection)	BCC
중복 감지(Duplicate Detection)	활성화(선택)
항목	**설명**
Protocol	DF-1 슬레이브(DF-1 Slave)
스테이션 주소(Station Address)	컨트롤러 스테이션 주소 번호 입력
전송 다시 시도(Transmit Retries)	3
슬레이브 폴링 타임아웃(Slave Poll Timeout)	3000
EOT 억제(EOT Suppression)	비활성화(선택 해제)
에러 감지(Error Detection)	BCC
중복 감지(Duplicate Detection)	활성화(선택)
항목	**설명**
Protocol	DF-1 슬레이브(DF-1 Slave)
스테이션 주소(Station Address)	컨트롤러 스테이션 주소 번호 입력
저장 및 전달 활성화(Enable Store and Forward)	저장 및 전달 태그를 사용하도록 상자 활성화(확인 표시)
에러 감지(Error Detection)	BCC

6. **확인**(OK)을 클릭합니다.

블록 전송

블록 전송 메세지의 경우 컨트롤러의 I/O 구성에 다음 모듈을 추가합니다.

블록 전송에 사용할 네트워크:	I/O 구성에 추가할 모듈:
ControlNet	로컬 통신 모듈(예: 1756-CNB 모듈) 원격 어댑터 모듈(예: 1771-ACN 모듈)
범용 원격 I/O	로컬 통신 모듈(예: 1756-DHRIO 모듈) 섀시의 각 랙 또는 랙 일부에 대해 원격 어댑터 모듈 1 개(예: 1771-ASB 모듈) 블록 전송 모듈(옵션)

통신 방법 또는 모듈 주소 지정

다음 표를 사용하여 메세지의 통신 방법 또는 모듈 주소를 선택합니다.

대상 장치	선택	지정	
LOGIX 5000 컨트롤러	CIP	다른 사양이 필요하지 않습니다.	
EtherNet/IP 네트워크를 통한 PLC-5 컨트롤러			
ControlNet 네트워크를 통한 PLC-5 컨트롤러			
SLC 5/05 컨트롤러			
DH+ 네트워크를 통한 PLC-5 컨트롤러	DH+	Channel	DH+ 네트워크에 연결된 1756-DHRIO 모듈의 채널 A 또는 채널 B.
DH+ 네트워크를 통한 SLC 컨트롤러		소스 링크	1756-DHRIO 모듈의 라우팅 테이블에서 컨트롤러 백플레인에 할당된 링크 ID. 라우팅 테이블의 소스 노드는 자동으로 컨트롤러의 슬롯 번호입니다.
PLC-3 프로세서		대상 링크	대상 장치가 있는 원격 DH+ 링크의 링크 ID.
PLC-2 프로세서		대상 노드	대상 장치의 스테이션 주소(8 진수).

		DH+ 링크 한 개만 있고 RSLinx Classic 소프트웨어를 사용하여 원격 링크용 DH/RIO 모듈을 구성하지 않는 경우, 소스 링크와 대상 링크 모두 0 을 지정합니다.	
RSLinx Classic 또는 FactoryTalk Linx 소프트웨어를 사용하여 EtherNet/IP 또는 ControlNet 네트워크를 통해 라우팅된 요청하지 않은 메세지를 수신하는 워크스테이션의 응용 프로그램	소스 ID 포함 CIP 선택할 경우 응용 프로그램이 컨트롤러의 데이터를 수신합니다.	소스 링크	RSLinx Classic 소프트웨어 항목의 원격 ID 또는 FactoryTalk Linx 의 바로 가기.
		대상 링크	RSLinx Classic 또는 FactoryTalk Linx 소프트웨어에서 설정된 가상 링크 ID(0 ~ 65535).
		대상 노드	응용 프로그램에서 RSLinx Classic 또는 FactoryTalk Linx 에 제공한 대상 ID(0 ~ 77 8 진수). RSLinx Classic 의 DDE 항목에는 77 을 사용합니다.
		ControlLogix Controller 의 슬롯 번호가 소스 노드로 사용됩니다.	
범용 원격 I/O 네트워크를 통한 블록 전송 모듈	RIO	Channel	RIO 네트워크에 연결된 1756-DHRIO 모듈의 채널 A 또는 채널 B.
		랙	모듈의 랙 번호(8 진수).
		그룹(Group)	모듈의 그룹 번호.
		슬롯	모듈의 슬롯 번호.
ControlNet 네트워크를 통한 블록 전송 모듈	ControlNet	슬롯	모듈의 슬롯 번호.

캐시 옵션 선택

MSG 명령어 구성에 따라 연결을 사용하여 데이터를 전송하거나 수신할 수 있습니다.

메세지 유형:	**통신 방법:**	**연결 사용 여부:**
CIP 데이터 표 읽기 또는 쓰기		사용자 선택(1)
PLC-2, PLC-3, PLC-5 또는 SLC(모든 유형)	CIP 소스 ID 포함 CIP	
	DH+	X
CIP 일반		사용자 선택(2)
블록 전송 읽기 또는 쓰기		X

1. CIP 데이터 표 읽기 또는 쓰기 메세지는 연결 또는 연결 안 됨 상태일 수 있습니다. Rockwell Automation 은 대부분의 응용 프로그램에 대해 CIP 데이터 표 읽기 또는 쓰기 메세지를 연결된 상태로 그대로 두기를 권합니다.

2. CIP 일반 메세지의 경우 연결됨 또는 연결 안 됨 상태로 될 수 있습니다. 그렇지만 대부분의 응용 프로그램에 대해 CIP 일반 메세지를 연결 안 됨 상태로 그대로 두기를 권합니다.

MSG 명령어에서 연결을 사용하는 경우 연결이 열린 채로 그대로 두거나(캐시) 메세지 전송을 마쳤을 때 연결을 닫도록 선택할 수 있습니다.

실행할 작업:	그러면:
연결 캐싱	MSG 명령어가 완료된 후 연결이 계속 열린 채로 있습니다. 따라서 실행 시간이 최적화됩니다. 메세지가 실행될 때마다 연결을 열면 실행 시간이 늘어납니다.
연결 캐싱 안 함	MSG 명령어가 완료된 후 연결이 닫힙니다. 따라서 다른 사용에 대해 그 연결을 사용할 수 있습니다.

컨트롤러마다 캐싱 가능한 연결의 수에 대한 제한이 있습니다.

사용한 컨트롤러:	캐싱 가능한 연결:
CompactLogix 5370 또는 ControlLogix 5570	최대 32 개의 연결.
ControlLogix 5580	최대 256 개의 연결.

여러 개의 메세지가 동일한 장치로 전송되는 경우 메세지 간에 연결을 공유할 수 있습니다.

MSG 명령어 적용 대상:	현재 상태:	그러면:
다른 장치		MSG 명령어당 1 개의 연결을 사용합니다.
동일 장치	동시에 활성화됨	MSG 명령어당 1 개의 연결을 사용합니다.
	동시에 활성화되지 않음	MSG 명령어는 연결 및 캐싱 버퍼를 각 1 개씩 사용합니다. 연결 및 버퍼를 공유합니다.

팁: 컨트롤러가 블록 전송 읽기 및 동일한 모듈에 블록 전송 쓰기 메세지 간에 전환되는 경우 연결을 구성하려면 두 메세지 모두 하나의 연결로 카운팅됩니다. 두 개의 메세지를 캐싱하면 캐시 리스트에서는 하나로 카운팅됩니다.

가이드라인

MSG 명령어를 계획 및 프로그래밍할 때는 아래 가이드라인을 따르십시오.

가이드라인	세부 정보
MSG 명령어별로 제어 태그를 만듭니다.	MSG 명령어별로 제어 태그가 필요합니다.
	데이터 유형 = MESSAGE
	범위 = 컨트롤러
	태그는 배열 또는 사용자 정의 데이터 유형에 포함될 수 없습니다.
소스 및/또는 대상 데이터를 컨트롤러 범위에서 유지합니다.	MSG 명령어는 컨트롤러 태그(Controller Tags) 폴더(컨트롤러 범위)에 있는 태그만 액세스할 수 있습니다.
16 비트 정수를 사용하는 장치에 MSG 를 적용하는 경우 프로젝트 전반에서 MSG 및 DINT 에서 INT 버퍼를 사용합니다.	메세지가 16 비트 정수를 사용하는 장치(예: PLC-5 또는 SLC 500 컨트롤러)로 전송되며 정수(REAL 아님)를 전송하는 경우 프로적트 전반에서 메세지 및 DINT 에 INT 버퍼를 사용합니다.
	Logix 컨트롤러는 32 비트 정수(DINT)로 작동할 때 보다 효율적으로 실행되고 메모리도 적게 소모되므로 프로젝트 효율성이 증가합니다.
	INT 와 DINT 사이에서 전환하려면 Logix 5000 Controllers Common Procedures Programming Manual (발행 번호 1756-PM001)을 참조하십시오.
자주 실행되는 연결된 MSG 를 캐싱합니다.	컨트롤러 수정 버전에 대해 허용되는 최대 한도까지 자주 실행되는 MSG 명령어 연결을 캐싱합니다. 캐싱할 경우 메세지가 실행될 때마다 컨트롤러가 연결을 열지 않아도 되므로 실행 시간이 최적화됩니다.
CompactLogix 5370 또는 ControlLogix 5570 컨트롤러의 경우 17 개 이상의 MSG 명령어를 동시에 활성화하려면 관리 전략을 사용하십시오. ControlLogix 5580 컨트롤러의 경우 257 개 이상의 MSG 명령어를 동시에 활성화하려면 관리 전략을	CompactLogix 5370 또는 ControlLogix 5570 컨트롤러의 경우 17 개 이상의 MSG 를 동시에 활성화하는 경우 일부 MSG 명령어가 지연으로 인해 큐에 들어갈 수 있습니다. ControlLogix 5580 컨트롤러의 경우 257 개 이상의 MSG 를 동시에 활성화하는 경우 일부 MSG 명령어가 지연으로 인해 큐에 들어갈 수 있습니다. 각 메세지가 실행되도록 하려면 아래 옵션 중 하나를 사용합니다.

사용하십시오.	각 메세지를 순차적으로 활성화합니다.
	메세지를 그룹 단위로 활성화합니다.
	여러 장치와 통신하도록 메세지를 프로그래밍합니다. 자세한 내용은 LOGIX 5000 Controllers Common Procedures Programming Manual (발행 번호 1756-PM001)을 참조하십시오.
	메세지 실행을 조정하는 로직을 프로그래밍합니다. 자세한 내용은 LOGIX 5000 Controllers Common Procedures Programming Manual (발행 번호 1756-PM001)을 참조하십시오.
(CompactLogix 5370 또는 ControlLogix 5570 컨트롤러에만 해당하는 내용) 연결되지 않고 캐싱되지 않은 MSG 의 수를 연결되지 않은 버퍼의 한도 미만으로 유지합니다.	컨트롤러는 10 ~ 40 개의 연결되지 않은 버퍼를 가질 수 있습니다. CompactLogix 5370 또는 ControlLogix 5570 컨트롤러의 경우 기본 개수는 10 입니다.
	명령어이 메세지 큐를 나갈 때 모든 연결되지 않은 버퍼가 사용 중이라면 명령어에 에러가 발생하여 데이터가 전송되지 않습니다.
	연결되지 않은 버퍼의 수를 증가할 수 있지만(최대 40 개) 지침 5 를 계속 준수합니다.
	연결되지 않은 버퍼의 수를 늘리려면 LOGIX 5000 Controllers Common Procedures Programming Manual (발행 번호 1756-PM001)을 참조하십시오.

SLC 메세지 지정하기

SLC 및 MicroLogix 컨트롤러와 통신하려면 SLC 메세지 유형을 사용합니다. 다음 표에는 명령어로 액세스 가능한 데이터 유형을 보여줍니다. 또한 표에서는 해당 LOGIX 5000 데이터 유형도 나타냅니다.

SLC 또는 MicroLogix 데이터 유형:	사용할 LOGIX 5000 데이터 유형:
F	REAL
L(MicroLogix 1200 및 1500 컨트롤러)	DINT
N	INT

블록 전송 메세지 지정하기

블록 전송 메세지 유형은 범용 원격 I/O 네트워크를 통해 블록 전송 모듈과 통신하는 데 사용됩니다.

목적:	선택해야 하는 명령:
블록 전송 모듈의 데이터 읽기 BTR 명령어를 대체하는 메세지 유형	블록 전송 읽기

블록 전송 모듈에 데이터 쓰기 BTW 명령어를 대체하는 메세지 유형	블록 전송 쓰기

블록 전송 메세지를 구성하려면 다음 가이드라인을 따르십시오.

- MESSAGE, AXIS 및 MODULE 구조를 제외하고 소스(BTW 용) 및 대상(BTR 용) 태그의 크기가 요청 데이터를 수용하기에 충분할 만큼 커야 합니다.
- 송수신할 16 비트 정수(INT)의 개수를 지정합니다. 0 ~ 64 개의 정수를 지정할 수 있습니다.

  팁: 블록 전송 모듈에서 전송할 16 비트 정수의 개수를 결정하게 하거나(BTR) 컨트롤러에서 64 비트 정수를 전송하게 하려면(BTW) 요소 수에 **0** 을 입력합니다.

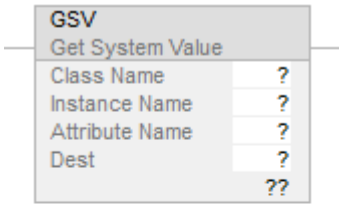
시스템 값 가져오기(GSV) 및 시스템 값 설정(SSV)

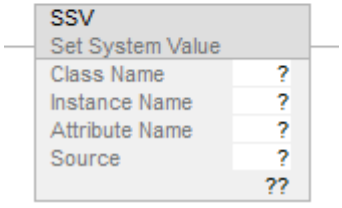
이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

GSV/SSV 명령어는 객체에 저장된 컨트롤러 시스템 데이터를 가져오고 설정합니다.

사용 가능한 언어

래더 다이어그램





펑션 블록

이 명령어는 펑션 블록에 사용할 수 없습니다.

ST(스트럭처드 텍스트)

GSV(ClassName,InstanceName,AttributeName,Dest)

SSV(ClassName,InstanceName,AttributeName,Source)

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
클래스 이름		이름	객체 클래스 이름
Instance name		이름	객체에 이름이 필요한 경우, 특정 객체의 이름
속성 이름		이름	객체의 속성 선택한 속성에 따라 데이터 유형이 달라집니다.
Destination (GSV)	SINT INT DINT REAL 구조	태그	속성 데이터의 대상
Source (SSV)	SINT INT DINT REAL 구조	태그	속성에 복사할 데이터가 포함된 태그

설명

GSV/SSV 명령어는 객체에 저장된 컨트롤러 상태 데이터를 가져오고 설정합니다. 컨트롤러는 객체에 상태 데이터를 저장합니다. PLC-5 프로세서와 마찬가지로 상태 파일이 없습니다.

GSV 명령어는 참인 경우 지정된 정보를 검색하여 대상에 저장합니다. SSV 명령어는 참인 경우 소스의 데이터를 사용하여 지정된 속성을 설정합니다.

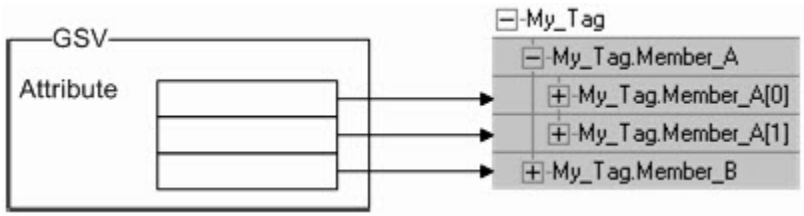
GSV/SSV 명령어를 입력하면 프로그래밍 소프트웨어에 명령어별로 유효한 객체 클래스, 객체 이름 및 속성 이름이 표시됩니다. GSV 명령어의 경우 모든 속성에 대한 값을 가져올 수 있습니다. SSV 명령어의 경우 소프트웨어에 설정 가능한 속성만 표시됩니다(SSV).

주의: SSV 명령어를 주의하여 사용하십시오. 객체를 변경하면 예기치 않은 컨트롤러 작동이나 직원 부상이 발생할 수 있습니다.

명령어로 인해 변경하지 않을 데이터가 변경되지 않는지 테스트 및 확인해야 합니다.

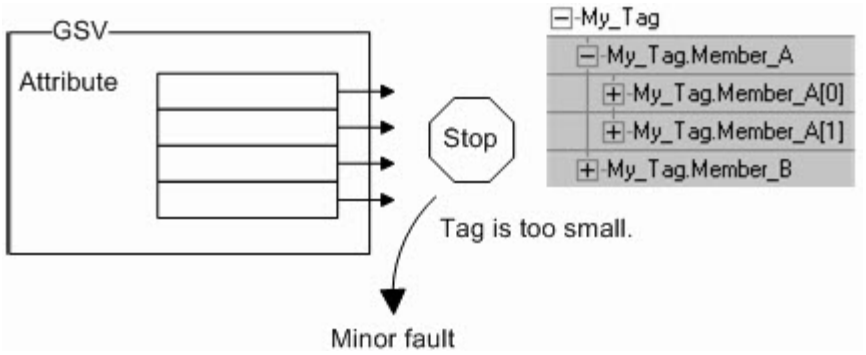
SSV 명령어는 태그의 다른 구성원에 구성원을 쓰고 GSV 명령어는 그 값을 읽습니다. 태그가 너무 작은 경우 명령어는 데이트를 쓰거나 읽지 못합니다. 대신 마이너 폴트가 기록합니다.

예 1



Member_A 가 속성에 비해 너무 작습니다. 따라서 GSV 명령어는 마지막 값을 Member_B 에 씁니다.

예 2



My_Tag 가 속성에 비해 너무 작습니다. 따라서 GSV 명령어는 중지되고 마이너 폴트가 로깅됩니다. Destination 태그는 그대로 변경되지 않을 채로 있습니다.

GSV/SSV 객체가 각 객체의 속성 및 관련 데이터 유형을 정의합니다. 예를 들어 프로그램 객체의 MajorFaultRecord 속성은 DINT[11] 데이터 유형을 사용해야 합니다.

연산 상태 플래그에 영향

아니요.

메이저/마이너 폴트

마이너 폴트 발생 조건:	폴트 유형	폴트 코드
객체 주소가 잘못되었습니다.	4	5
지정된 객체가 GSV/SSV 를 지원하지 않습니다.	4	6
속성이 유효하지 않습니다.	4	6
SSV 명령어에 충분한 정보가 제공되지 않았습니다.	4	6
GSV 대상이 요청된 데이터를 저장할 만큼 크지 않습니다.	4	7

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

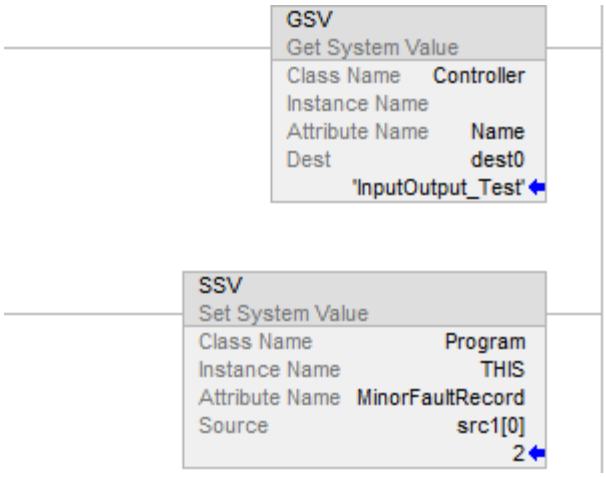
Condition	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

Condition	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참임을 참조하십시오.
사후 스캔	래더 다이어그램 표의 사후 스캔을 참조하십시오.

예

래더 다이어그램



ST(스트럭처드 텍스트)

```
GSV (Program,THIS,LASTSCANTIME,dest1);

SSV (Program, THIS, MinorFaultRecord, src[0]);
```

추가 참조

데이터 변환 페이지의 967

공통 특성 페이지의 963

GSV/SSV 객체 페이지의 230

GSV/SSV 안전 객체 페이지의 290

GSV/SSV 프로그래밍 예 페이지의 226

즉시 출력(IOT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

IOT 명령어는 지정된 출력 데이터(I/O 모듈의 출력 태그 또는 산출된 태그)를 즉시 업데이트합니다. IOT 명령어를 활성화하여 실행하려면 모듈과의 연결이 열려 있어야 합니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

IOT (output_tag)

피연산자

래더 다이어그램

피연산자	유형	형식	설명
Update Tag		태그	업데이트할 속성 태그에 복사할 데이터가 들어 있는 태그. 다음 중 하나: I/O 모듈의 출력 태그 또는 산출된 태그

ST(스트럭처드 텍스트)

래더 다이어그램 IOT 명령어와 피연산자가 같습니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 ST(스트럭처드 텍스트) 구문을 참조하십시오.

설명

IOT 명령어는 출력 연결의 RPI 를 무시하고 연결을 통해 최신 데이터를 전송합니다.

출력 연결은 I/O 모듈의 출력 태그 또는 산출된 태그와 관련된 연결입니다. 산출된 태그의 연결인 경우 IOT 명령어는 또한 컨수머 컨트롤러에 이벤트 트리거를 전송합니다. 그러면 IOT 명령어는 사용 컨트롤러에서 이벤트 태스크를 트리거할 수 있습니다.

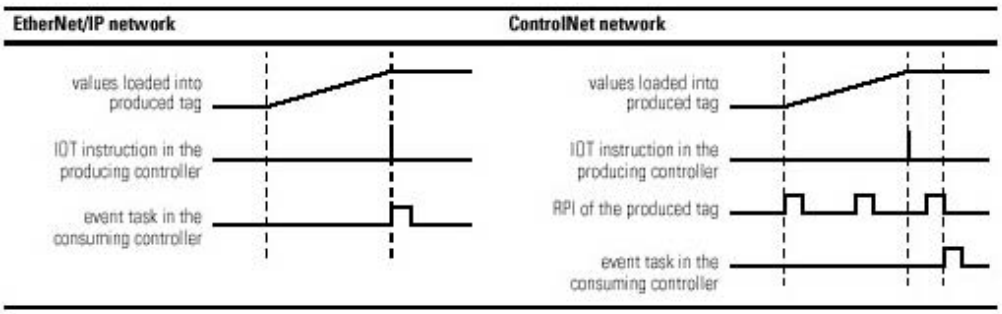
IOT 명령어 및 산출된 태그를 사용하여 컨수머 컨트롤러에서 이벤트 태스크를 트리거하려면 **태그 속성**(Tag Properties) 대화 상자의 연결(Connection) 탭에서 IOT 명령어를 통해 프로그래밍 방식으로 컨수머에게 이벤트 트리거 전송(Programmatically (IOT Instruction) Send Event Trigger to Consumer) 확인란을 선택합니다.

팁: CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러의 경우, 원격 네트워크를 통해 5069 I/O 를 제어할 때 네트워크를 통해 전송하기 위해 동일한 RPI 속도로 구성된 모듈 연결을 하나의 패킷으로 그룹화하는 데 최적화가 사용됩니다. 이러한 태그 중 하나에서 IOT 가 사용되는 경우 IOT 는 동일한 RPI 및 백플레인에서 구성되고 해당 태그와 함께 그룹화되는 다른 모듈의 일부 데이터 태그를 즉시 업데이트할 수 있습니다. 이러한 업데이트가 필요 없는 경우에는 RPI 를 다른 모듈 연결의 RPI 와 정확히 같지 않도록 만들어서 업데이트를 방지할 수 있습니다.

컨트롤러 간 네트워크 유형은 사용 컨트롤러가 IOT 명령어를 통해 새 데이터 및 이벤트 트리거를 수신하는 시점을 결정합니다.

이 네트워크를 통해	사용 장치에서 데이터 및 이벤트 트리거 수신
백플레인	즉시
EtherNet/IP	즉시
ControlNet	사용된 태그(연결)의 API 내에서

다음 다이어그램에서는 EtherNet/IP 와 ControlNet 네트워크에서 IOT 명령어를 통한 데이터 수신을 비교합니다.



연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 지정된 태그의 연결을 업데이트하고 연결의 RPI 타이머를 리셋합니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	N/A
정상 실행	래더 다이어그램에서 렁-입력-조건이 참임을 참조하십시오.
사후 스캔	N/A

예

IOT 명령어가 실행되면 Local:5:0 태그 값이 출력 모듈로 즉시 전송됩니다.

래더 다이어그램

ST(스트럭처드 텍스트)

IOT (Local:5:0),

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

시스템 값에 액세스

이 절차를 수행하면 LOGIX 5000 컨트롤러에 대한 상태 정보를 가져오거나 사용할 수 있습니다.

실행할 작업:	참조할 도움말 항목:
논리에 특정 키워드를 사용하여 특정 이벤트 모니터링	상태 플래그 모니터링 페이지의 297
시스템 값 가져오기 또는 설정	시스템 데이터 가져오기 및 설정 페이지의 224
컨트롤러의 메모리에 대한 정보 가져오기	컨트롤러 메모리 정보 알아내기 페이지의 217

컨트롤러 메모리 정보 알아내기

컨트롤러 메모리는 I/O 메모리와 확장 메모리로 구분됩니다. 다음 표에서는 각 메모리 유형별 컨트롤러 사용 방법을 보여줍니다.

대상	사용 메모리
I/O 태그	I/O 메모리
생성 태그	
사용 태그	
MSG 명령어를 통한 통신	
워크스테이션에서 통신	
I/O, 생산 또는 사용 이외의 태그	확장 메모리
로직 루틴	
RSLinx Classic 를 사용하는 폴링된 (OPC/DDE) 태그에서 통신.	I/O 메모리 및 확장 메모리

컨트롤러는 32 비트 워드로 값을 반환합니다. 바이트 단위로 값을 보려면 4 를 곱하기만 하면 됩니다.

이 절차를 사용하여 컨트롤러 메모리에 대한 다음 정보를 가져옵니다.

- 사용 가용한(가용) I/O 및 확장 메모리
- 총 I/O 및 확장 메모리
- 최대 I/O 및 확장 메모리 인접 블록

컨트롤러에서 메모리 정보 가져오기

컨트롤러에서 메모리 정보를 가져오려면 다음과 같이 구성되는 메세지(MSG) 명령어를 실행합니다.

메세지 속성(Message Properties) 대화 상자에서 - 구성(Configuration) 탭:

For this item:	Type or select:	Which means:	
Message Type	CIP Generic	Execute a Control and Information Protocol command.	
Service Type	Custom	Create a CIP Generic message that is not available in the drop-down list.	
Service Code	3	Use the GetAttributeList service. This lets you read specific information about the controller.	
Class	72	Get information from the user memory object.	
Instance	1	This object contains only 1 instance.	
Attribute	0	Null value	
Source Element	*source_array* of type SINT[12]		
	In this element:	**Enter:**	**Which means:**
	source_array[0]	5	Get 5 attributes
	source_array[1]	0	Null value
	source_array[2]	1	Get free memory
	source_array[3]	0	Null value
	source_array[4]	2	Get total memory
	source_array[5]	0	Null value
	source_array[6]	5	Get largest contiguous block of additional free expansion memory
	source_array[7]	0	Null value
	source_array[8]	6	Get largest contiguous block of free I/O memory
	source_array[9]	0	Null value
	source_array[10]	7	Get largest contiguous block of free expansion memory
	source_array[11]	0	Null value
Source Length	12	Write 12 bytes (12 SINTs).	
Destination	*INT_array* of type INT[29]		

메세지 속성(Message Properties) 대화 상자에서 - 통신(Communication) 탭:

For this item:	Type:
Path	1, *slot_number_of_controller*

원하는 메모리 정보 선택

MSG 명령어가 다음 정보를 INT_array(MSG 명령어의 대상 태그)로 반환합니다.

중요: 1756-L55M16 컨트롤러의 경우, MSG 명령어가 각 확장 메모리 범주에 대해 2 개 값을 반환합니다. 1756-L55M16 컨트롤러의 가용 또는 총 확장 메모리를 정하려면 범주에 두 값을 더합니다.

If you want the:	Then copy these array elements:	Description:
amount of free I/O memory (32-bit words)	*INT_array*[3]	lower 16 bits of the 32 bit value
	INT_array[4]	upper 16 bits of the 32 bit value
amount of free expansion memory (32-bit words)	*INT_array*[5]	lower 16 bits of the 32 bit value
	INT_array[6]	upper 16 bits of the 32 bit value
1756-L55M16 controllers only—amount of additional free expansion memory (32-bit words)	*INT_array*[7]	lower 16 bits of the 32 bit value
	INT_array[8]	upper 16 bits of the 32 bit value
total size of I/O memory (32-bit words)	*INT_array*[11]	lower 16 bits of the 32 bit value
	INT_array[12]	upper 16 bits of the 32 bit value
total size of expansion memory (32-bit words)	*INT_array*[13]	lower 16 bits of the 32 bit value
	INT_array[14]	upper 16 bits of the 32 bit value
1756-L55M16 controllers only—additional expansion memory (32-bit words)	*INT_array*[15]	lower 16 bits of the 32 bit value
	INT_array[16]	upper 16 bits of the 32 bit value
1756-L55M16 controllers only—largest contiguous block of additional free expansion memory (32-bit words)	*INT_array*[19]	lower 16 bits of the 32 bit value
	INT_array[20]	upper 16 bits of the 32 bit value
largest contiguous block of free I/O memory (32-bit words)	*INT_array*[23]	lower 16 bits of the 32 bit value
	INT_array[24]	upper 16 bits of the 32 bit value
largest contiguous block of free expansion memory (32-bit words)	*INT_array*[27]	lower 16 bits of the 32 bit value
	INT_array[28]	upper 16 bits of the 32 bit value

INT 를 DINT 로 변환

MSG 명령어는 두 개의 별도 INT 로 각 메모리 값을 반환합니다.

- 첫 번째 INT 는 값 중에서 하위 16 비트를 나타냅니다.

- 두 번째 INT 는 값 중에서 상위 16 비트를 나타냅니다.

별도 INT 를 사용 가능한 한 값으로 변환하려면 복사(COP) 명령어를 사용합니다.

피연산자:	지정 내용:	의미:
소스(Source)	두 개의 요소 쌍 중 첫 번째 INT(하위 16 비트)	하위 16 비트로 시작
대상(Destination)	32 비트 값을 저장할 DINT 태그	값을 DINT 태그로 복사
길이(Length)	1	1 과 대상 데이터 유형에 바이트 수를 곱한 값을 복사합니다. 이 경우, 명령어는 4 바이트(32 비트)를 복사하는데, 16 비트의 하위 값과 상위 값을 한 개의 32 비트 값으로 결합합니다.

DeviceNet 상태 코드

다음은 DeviceNet 상태 코드입니다.

상태 코드	상태 설명	권장 작업
0 ~ 63	스캐너 또는 슬레이브 장치의 DeviceNet 노드 주소.	없음.
65	AutoScan 옵션이 켜져 있으며 스캐너가 유휴 모드입니다.	없음.
67	스캐너가 보조 스캐너입니다.	없음.
68	기본 스캐너가 보조 스캐너를 감지하지 못했습니다.	다른 스캐너를 보조 스캐너로 구성합니다.
69	기본 및 보조 구성이 일치하지 않습니다.	보조 스캐너 구성을 확인하십시오.
70	스캐너 주소가 이미 네트워크에 있는 다른 장치에 의해 사용 중입니다.	스캐너 주소를 사용하지 않는 주소로 변경합니다.
71	스캔 목록에 있는 유효하지 않은 데이터.	RSNetWorx 소프트웨어를 사용하여 스캔 목록을 재구성합니다.

72	슬레이브 장치가 통신 중지됩니다. 다음 시도 중에 슬레이브 장치에서 통신을 재설정하지 않으면 상태 코드가 78 로 변경됩니다.	• 슬레이브 장치 전원과 네트워크 연결을 확인하십시오. • 슬레이브 장치가 폴링되는 경우, 슬레이브 장치가 데이터를 반환하기 위한 인터스캔 지연이 충분한지 확인하십시오. • 슬레이브 장치가 적절하게 작동하는지 확인하십시오.
73	슬레이브 장치의 ID 정보가 스캐너의 전자 키와 일치하지 않습니다.	• 올바른 슬레이브 장치가 이 주소에서 연결되어 있는지 확인하십시오. • 슬레이브 장치가 지정된 전자 키(공급업체, 제품 코드, 제품 유형)와 일치하는지 확인하십시오. • 슬레이브 장치가 적절하게 작동하는지 확인하십시오.
74	스캐너가 DeviceNet 통신 포트의 데이터 오버런을 감지했습니다.	• 네트워크 통신 트래픽을 확인하십시오. • 슬레이브 장치가 적절하게 작동하는지 확인하십시오.
75	다음 중 하나 또는 둘 다 있습니다. • 스캐너에 스캐너 목록이 없습니다. • 스캐너가 다른 장치로부터 통신을 수신하지 못했습니다.	스캐너에 다음 사항이 있는지 확인하십시오. • 구성된 스캔 목록 • 네트워크에 적절한 연결
76	스캐너에 직접 네트워크 트래픽이 없습니다. 스캐너가 다른 네트워크 통신은 수신하지만 직접 전달되는 통신은 수신하지 않습니다.	없음.
77	초기화하는 동안 슬레이브 장치가 예상하는 데이터 크기가 해당 스캔 목록 항목의 크기와 일치하지 않습니다.	• 슬레이브 장치의 입력 및 출력 크기가 정확한지 RSNetWorx 소프트웨어를 사용하여 슬레이브 장치와 스캔 목록을 확인하십시오. • 슬레이브 장치가 적절하게 작동하는지 확인하십시오.

78	슬레이브 장치가 스캔 목록에 구성되어 있지만 통신하지 않습니다.	• 슬레이브 장치 전원과 네트워크 연결을 확인하십시오.
		• 슬레이브 장치가 폴링되는 경우, 슬레이브 장치가 데이터를 반환하기 위한 인터스캔 지연이 충분한지 확인하십시오.
		• 필요한 경우, RSNetWorx 소프트웨어를 사용하여 다음을 수행합니다.
		  • 슬레이브 장치를 DeviceNet 네트워크에 추가합니다.
		  • 스캐너 스캔 목록에서 슬레이브 장치를 삭제합니다.
		  • 스캐너 스캔 목록에서 슬레이브 장치를 금지합니다.
		• 슬레이브 장치가 적절하게 작동하는지 확인하십시오.

79	스캐너가 메시지를 전송하지 못했습니다.	• 스캐너가 유효 네트워크에 연결되어 있는지 확인하십시오. • 연결이 끊어진 케이블이 있는지 확인하십시오. • 네트워크 전송 속도를 확인하십시오.
80	스캐너가 유휴 모드입니다.	원하는 경우, 다음을 수행하여 스캐너를 실행 모드에 둡니다. • 컨트롤러에 있는 키 스위치를 사용하거나 Logix Designer 응용 프로그램을 통해 컨트롤러를 실행/원격 실행 모드에 둡니다. • 스캐너의 O.CommandRegister.Run 비트를 켭니다.
81	컨트롤러가 스캐너를 폴트 모드로 설정했습니다.	스캐너의 O.CommandRegister.Run 비트가 켜집니다. 컨트롤러가 이 비트를 설정하는 원인이 되는 조건을 수정한 다음 이 비트를 끕니다.

82	슬레이브 장치의 조각난 I/O 메시지 시퀀스에서 오류가 감지됩니다.	• RSNetWorx 소프트웨어를 사용하여 다음을 수행합니다.
		  • 슬레이브 장치에 대해 스캔 목록 항목을 점검하여 입력 및 출력 데이터 크기가 정확한지 확인하십시오.
		  • 슬레이브 장치 구성을 확인하십시오.
		• 슬레이브 장치가 적절하게 작동하는지 확인하십시오.

83	스캐너가 슬레이브 장치와 통신하려고 할 때 슬레이브 장치가 에러 응답을 반환합니다.	• RSNetWorx 소프트웨어를 사용하여 다음을 수행합니다.
		  • 스캔 목록이 정확한지 확인하십시오.
		  • 슬레이브 장치 구성을 확인하십시오. 슬레이브 장치가 다른 스캐너의 스캔 목록에 있을 수 있습니다.
		• 슬레이브 장치에 대해 전원을 껐다 켭니다.
		• 슬레이브 장치가 적절하게 작동하는지 확인하십시오.

84	스캐너가 DeviceNet 네트워크를 초기화하는 중입니다.	없음. 스캐너가 네트워크에 있는 모든 슬레이브 장치를 초기화하려고 할 때 이 코드가 지워집니다.
85	런타임 중에 슬레이브 장치에서 보낸 데이터 크기가 해당 스캔 목록 항목에 있는 크기와 일치하지 않습니다.	변수 길이 폴 데이터가 지원되지 않기 때문에 슬레이브 장치가 적절하게 작동하는지 확인하십시오.
86	슬레이브 장치가 유휴 모드이거나 스캐너가 실행 모드일 때 데이터를 생성하지 않습니다.	• 슬레이브 장치의 구성 및 상태를 확인하십시오. • 두 개의 스캐너를 마스터/슬레이브 관계로 설정할 경우 해당 스캐너가 실행 모드인지 확인하십시오.
87	소유 스캐너가 슬레이브 스캐너와의 통신을 설정하지 않았기 때문에 스캐너가 슬레이브 장치에서 공유한 입력을 수신할 수 없습니다.	• 소유 스캐너 연결 및 구성을 확인합니다. • 슬레이브 장치가 데이터를 생성하지 않을 수 있습니다.
88	슬레이브 장치에 대한 I/O 파라미터(예를 들어, 폴링 또는 스트로브, 전자 키, 데이터 크기)가 이 스캐너와 소유 스캐너 사이에 다르게 구성되어 있기 때문에 스캐너는 슬레이브 장치의 공유 입력을 수신할 수 없습니다.	이 스캐너의 파라미터가 소유 스캐너의 파라미터와 일치하도록 공유 입력 스캔 목록 항목에 대한 I/O 파라미터를 다시 구성합니다.
89	스캐너는 자동 장치 복구(ADR) 파라미터를 사용하여 슬레이브 장치를 구성하지 못했습니다.	호환 슬레이브 장치를 설치했는지 확인하십시오.
90	컨트롤러가 스캐너를 비활성 모드로 설정했습니다.	필요한 경우, 스캐너에 대해 O.CommandRegister.DisableNetwork 비트를 꺼서 스캐너를 활성화합니다.

91	버스오프 상태는 케이블 또는 신호 에러로 인해 가져옵니다.	• 스캐너, 슬레이브 장치 및/또는 네트워크에 대해 전원을 껐다 켭니다.
		• 모든 장치에서 전송 속도가 동일하게 설정되어 있는지 확인하십시오.
		• DeviceNet 케이블 연결을 점검하여 CAN(파란색과 흰색) 배선과 전원 또는 실드(검정색, 빨간색, 보호색) 배선 사이에 단락이 없는지 확인하십시오.
		• 다음 노이즈 소스에 미디어 시스템을 확인하십시오.
		  • 장치가 고전압 전원 케이블 근처에 배치됨
		  • 잘못되거나 중단 저항기가 없음
		  • 접지가 적절하지 않음
		• 네트워크에 있는 노이즈를 생성하는 장치 또는 잘못된 데이터

92	DeviceNet 케이블이 스캐너 통신 포트에 전원을 공급하지 않습니다.	• 네트워크의 24V 직류 전원 공급 장치가 제대로 작동하는지 확인하십시오. • 케이블 상태가 양호한지 확인하십시오. • 스캐너에 대한 케이블 연결을 확인하십시오.
95	스캐너 펌웨어가 업데이트 중이거나 구성을 다운로드하는 중입니다.	없음. 업데이트가 진행 중인 동안에는 스캐너 연결을 끊지 마십시오. 그렇지 않으면 스캐너 메모리에 있는 기존 데이터가 소실됩니다.
97	컨트롤러가 스캐너를 중단 모드에 두었습니다.	스캐너의 O.CommandRegister.HaltScanner 비트가 켜져 있습니다. 이 비트를 끈 다음 스캐너 전원을 껐다 켭니다.
98	일반 펌웨어 에러.	장치를 교체합니다.
99	시스템 실패.	장치를 교체합니다.

시스템 데이터 가져오기 및 설정

컨트롤러가 객체에서 시스템 데이터를 저장합니다. PLC-5 컨트롤러에서와 같이 상태 파일이 없습니다. GSV/SSV 명령어를 사용하여 객체에 저장된 컨트롤러 시스템 데이터를 가져오고 설정합니다.

- GSV 명령어가 지정된 정보를 검색하여 대상에 둡니다.
- SSV 명령어가 소스 데이터롤 통해 지정된 속성을 설정합니다.

주의: SSV 명령어를 신중하게 사용합니다. 객체를 변경하면 예기치 않은 컨트롤러 작동이나 직원 부상이 발생할 수 있습니다.

시스템 값을 가져오거나 설정하려면:

1. Logix Designer 응용 프로그램 프로젝트를 엽니다.
2. **도움말**(Help) 메뉴에서, **내용**(Contents)을 클릭합니다.
3. **인덱스**(Index)를 클릭합니다.
4. **gsv/ssv 객체**(gsv/ssv objects)를 입력하고 **표시**(Display)를 클릭합니다.
5. 필요한 객체를 클릭합니다.

다음을 가져오거나 설정하려면	다음을 클릭함
서보 모듈 축	AXIS
시스템 오버헤드 타임 슬라이스	CONTROLLER
컨트롤러의 실제 하드웨어	CONTROLLERDEVICE
하나의 섀시에 있는 장치에 대한 조정 시스템 시간	CST
시리얼 포트에 대한 DF1 통신 드라이버(시리얼 포트가 있는 컨트롤러에만 해당)	DF1
컨트롤러의 폴트 기록	FAULTLOG
메세지 명령어 속성	MESSAGE
상태, 폴트, 통신 경로 및 모듈 모드	MODULE
축 그룹	MOTIONGROUP
프로그램의 폴트 정보 또는 스캔 시간	PROGRAM
루틴의 인스턴스 번호	ROUTINE
시리얼 포트 구성(시리얼 포트가 있는 컨트롤러에만 해당)	SERIALPORT
태스크 속성 또는 경과 시간	TASK
컨트롤러 벽시계 시간	WALLCLOCKTIME
컨트롤러의 시간 동기화 상태	TIMESYNCHRONIZE

6. 객체의 속성 목록에서 액세스하려는 속성을 식별합니다.

7. 속성 값에 대해 태그를 생성합니다.

속성의 데이터 유형	그러면
하나의 요소(예: DINT)	속성에 대해 태그를 생성합니다.
두 개 이상의 요소(예: DINT[7])	속성에서 사용되는 데이터 구성과 맞는 사용자 정의 데이터 유형을 생성합니다. 그런 다음 속성에 대해 태그를 생성하고 해당 데이터 유형을 사용합니다.

8. 래더 로직 루틴에서 적절한 명령어를 입력합니다.

실행할 작업	다음 명령어 입력
속성 값 가져오려면	GSV
속성 값 설정하려면	SSV

9. 필요한 피연산자를 명령어에 할당하려면

   해당 피연산자에 대한 자세한 내용은 GSV/SSV 명령어를 참조하십시오.

추가 참조

(GSV)(시스템 값 가져오기) 및 시스템 값 설정 페이지의 209

GSV/SSV 프로그래밍 예

다음 예에서는 GSV 명령어를 사용하여 폴트 정보를 가져옵니다.

예 1: I/O 폴트 정보 가져오기

이 예에서는 I/O 모듈인 disc_in_2 의 폴트 정보를 가져와서 사용자 정의 구조인 disc_in_2_info 에 데이터를 저장합니다.

래더 다이어그램

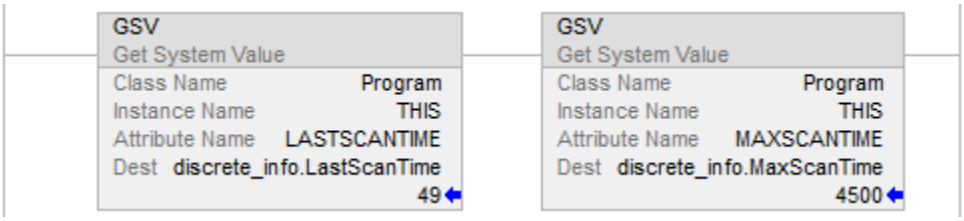
ST(스트럭처드 텍스트)

```
GSV(MODULE,disc_in_2,FaultCode,disc_in_2_info.FaultCode);
GSV(MODULE,disc_in_2,FaultInfo,disc_in_2_info.FaultInfo);
GSV(MODULE,disc_in_2,Mode,disc_in_2_info.Mode);
```

예 2: 프로그램 상태 정보 가져오기

이 예에서는 개별 프로그램의 상태 정보를 가져와서 사용자 정의 구조인 discrete_info 에 데이터를 저장합니다.

래더 다이어그램



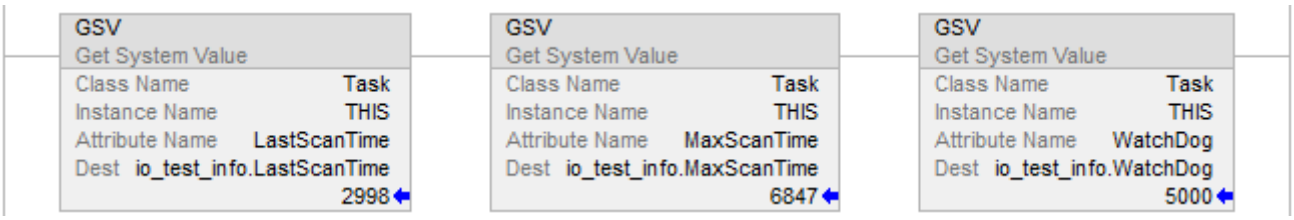
ST(스트럭처드 텍스트)

```
GSV(PROGRAM,DISCRETE,LASTSCANTIME,discrete_info.LastScanTime);
GSV(PROGRAM,DISCRETE,MAXSCANTIME,discrete_info.MaxScanTime);
```

예 3: 태스크 상태 정보 가져오기

이 예에서는 태스크 IO_test 의 상태 정보를 가져와서 사용자 정의 구조인 io_test_info 에 데이터를 저장합니다.

래더 다이어그램



ST(스트럭쳐드 텍스트)

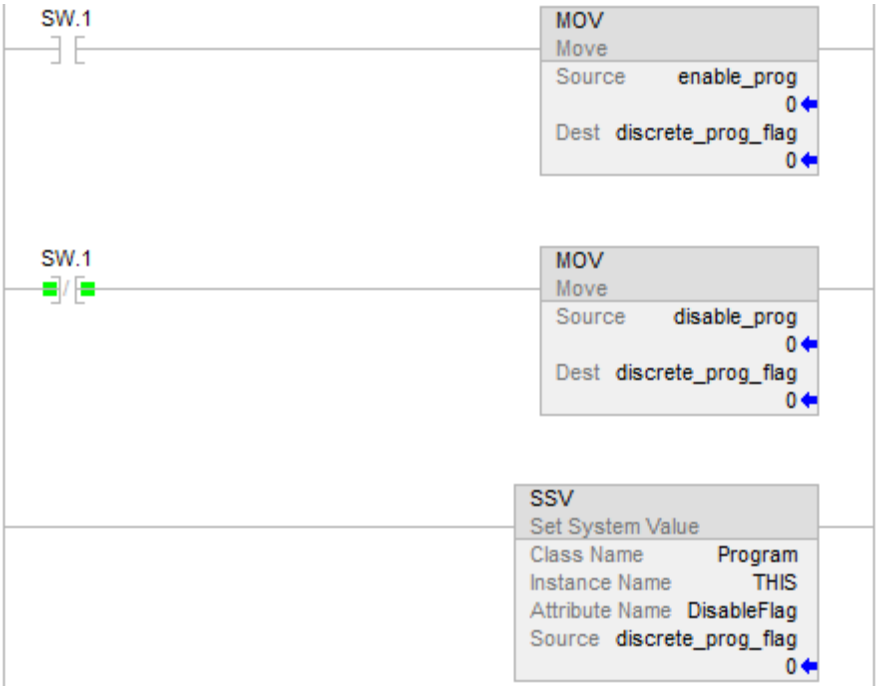
```
GSV(TASK,IO_TEST,LASTSCANTIME,io_test_info.LastScanTime);
GSV(TASK,IO_TEST,MAXSCANTIME,io_test_info.MaxScanTime);
GSV(TASK,IO_TEST,WATCHDOG,io_test_info.Watchdog);
```

활성화 및 비활성화 플래그 설정

다음 예에서는 SSV 명령어를 사용해 프로그램을 활성화하거나 비활성화합니다. 이 방법을 사용하여 I/O 모듈 또한 활성화하거나 비활성화할 수 있으며 PLC-5 프로세서에서 금지 비트를 사용하는 것과 유사한 프로그램 솔루션에 해당합니다.

SW.1 의 상태를 기준으로 개별 프로그램의 비활성화 플래그 속성에 해당하는 값을 설정합니다.

래더 다이어그램



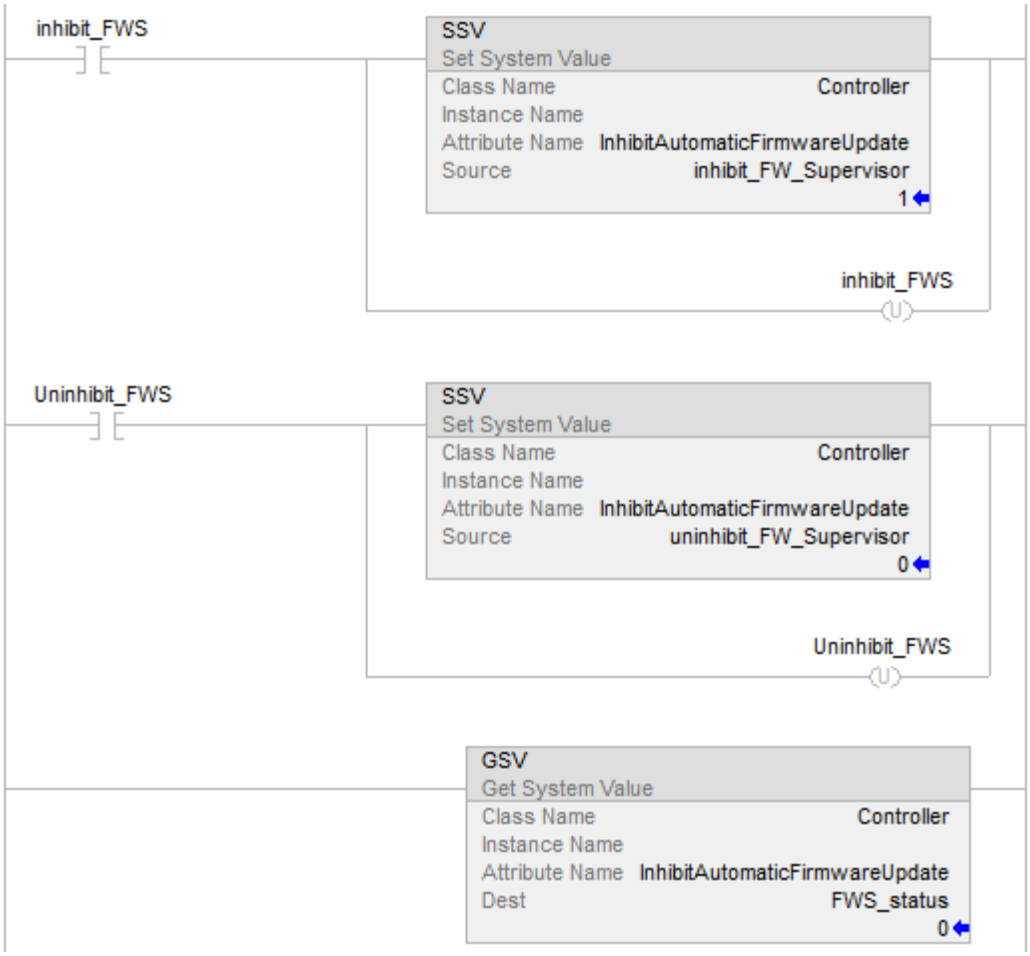
ST(스트럭처드 텍스트)

```
IF SW.1 THEN
discrete_prog_flag := enable_prog;
   ELSE
   discrete_prog_flag := disable_prog;
END_IF;
SSV(PROGRAM,DISCRETE,DISABLEFLAG,discrete_prog_flag);
```

FirmwareSupervisor 자동 펌웨어 업데이트 금지 및 금지 해제

다음 예에서는 GSV/SSV 명령어를 사용하여 컨트롤러의 자동 펌웨어 업데이트 속성을 금지하거나 금지 해제합니다. 값을 1 로 쓰면 기능을 금지합니다. 값을 0 으로 쓰면 기능이 금지 해제됩니다. GSV 명령어를 사용하여 속성의 상태를 읽을 수도 있습니다.

래더 다이어그램



GSV/SSV 객체

GSV/SSV 명령어를 입력할 때 액세스할 객체와 속성을 지정합니다. 경우에 따라 동일한 객체 유형의 인스턴스가 두 개 이상일 수 있습니다. 객체 이름을 지정합니다. 예를 들어, 모든 태스크에는 액세스 권한을 얻을 수 있도록 태스크 이름을 지정해야 하는 태스크 객체가 있습니다.

중요: GSV 명령어의 경우 데이터의 지정된 크기만 대상에 복사됩니다. 예를 들어, 속성을 SINT 로, 대상을 DINT 로 지정할 경우 DINT 대상의 하위 8 비트만 업데이트되며 남은 24 비트는 변경되지 않습니다.

중요: v21 펌웨어에서 알람을 울리도록 구독 기능에서 알람 버퍼가 제거되었으며 이제 더 이상 사용할 수 없습니다. 프로젝트를 확인하면 이전에 알람 버퍼 속성을 참조하던 GSV 명령어는 무효화됩니다. 이 속성에서 활용한 모든 응용 프로그그램 코드를 정확하게 변경하는 것은 프로그래머의 책임입니다.

GSV/SSV 객체입니다. 액세스 가능한 객체는 컨트롤러에 따라 결정됩니다.

- AddOnInstructionDefinition 페이지의 232
- 축 페이지의 236
- 컨트롤러 페이지의 248
- ControllerDevice 페이지의 250
- CoordinateSystem 페이지의 253
- CST 페이지의 258
- DF1 페이지의 261
- FaultLog 페이지의 265
- HardwareStatus 페이지의 266
- 메시지 페이지의 257
- 모듈 페이지의 268
- MotionGroup 페이지의 256
- 프로그램 페이지의 277
- 중복 페이지의 272
- 루틴 페이지의 271
- 안전 페이지의 278
- SerialPort 페이지의 280
- 태스크 페이지의 281
- TimeSynchronize 페이지의 283
- WallClockTime 페이지의 288

추가 참조

시스템 값 가져오기(GSV) 및 시스템 값 설정(SSV) 페이지의 209

입력/출력 명령어 페이지의 165

AddOnInstruction Definition 객체 액세스

AddOnInstructionDefinition 객체를 통해 일반적으로 사용되는 로직 집합에 대한 명령어를 사용자 지정하게 하고, 이러한 로직에 공통 인터페이스를 제공하고, 명령어에 대한 문서를 제공합니다.

자세한 내용은 LOGIX 5000 Controllers Add-On Instructions Programming Manual(발행 번호 1756-PM010)을 참조하십시오.

속성	데이터 유형	표준 태스크 내 명령어	안전 태스크 내 명령어	설명
LastEditDate	LINT	GSV	없음	애드온 명령어 정의에 대한 마지막 편집 날짜 및 타임스탬프.
MajorRevision	DINT	GSV	없음	애드온 명령어의 주 수정 버전 정보.
MinorRevision	DINT	GSV	없음	애드온 명령어의 부 수정 버전 정보.
Name	문자열	GSV	GSV	애드온 명령어의 이름.
RevisionExtendedText	문자열	GSV	없음	애드온 명령어의 수정 버전을 설명하는 텍스트.
SafetySignature ID	DINT	GSV	없음	안전 프로젝트에서 애드온 명령어 정의의 ID 번호, 날짜 및 타임스탬프.
SignatureID	DINT	GSV	없음	애드온 명령어 정의의 32 비트 식별 번호.
Vendor	문자열	GSV	없음	애드온 명령어를 생성한 공급업체.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

ALARMBUFFER 객체 액세스

ALARMBUFFER 객체는 게시자/구독자 인프라의 일부입니다. 게시자/구독자 인프라는 Logix 컨트롤러 통신 하위 시스템의 일부입니다. Logix 컨트롤러 통신 하위 시스템은 컨트롤러의 하위 시스템에서 보낸 메시지를 다른 장치에서 수신하도록 하는 CIP 에 대한 게시자/구독자 메시징 패턴을 구현합니다. 현재, 디지털 및 아날로그 알람과 배치 장비 단계 하위 시스템에서는

게시자/구독자 인프라를 사용하여 CIP 를 통해 메세지를 구독 응용 프로그램에 전달합니다.

ALARMBUFFER 객체를 사용하면 게시자/구독자 하위 시스템에 대한 연결이 존재하는지 확인하고 연결 상태를 확인할 수 있습니다. AlarmBuffer 객체 인스턴스는 모든 구독 응용 프로그램에 대해 존재합니다. 즉, AlarmBuffer 객체는 한 시점에서 존재할 수 있지만 다른 시점에는 존재하지 않을 수 있습니다. 따라서 시스템 값 가져오기(GSV) 명령어는 대상 태그(INT[0].0)의 일부로 상태를 반환합니다. 상태 비트가 0 인 경우 이는 대부분 AlarmBuffer 객체가 더 이상 존재하지 않음을 의미할 것입니다.

속성	데이터 유형	명령어	설명	
AlarmBufferInstance	DINT[n]	GSV	AlarmBuffer 객체 ID 를 반환함.	
			DINT[0]	AlarmBuffer 객체 수.
			DINT[1...(n-1)]	AlarmBuffer 객체 ID.
			AlarmBuffer 객체 수가 n-1 보다 크면 첫 번째 (n-1) 객체의 ID 만 반환됩니다. 이 속성에 대한 AlarmBuffer 인스턴스 ID 는 지정할 필요가 없습니다.	
AlarmBufferStatus	INT[2]	GSV	지정된 AlarmBuffer 객체의 상태를 반환함. 개별 인스턴스의 상태를 가져오려면 AlarmBuffer 인스턴스 ID 를 지정해야 합니다.	
			INT[0].0	1-AlarmBufferStatus 속성이 유효합니다. 0-AlarmBufferStatus 속성이 유효하지 않습니다.
			INT[1]	AlarmBuffer 상태 속성 값.
			상태 속성에는 다음이 포함되어 있습니다.	
			INT[1].0	1-다중 메세지 패킷이 활성화됩니다. 0-다중 메세지 패킷이 비활성화됩니다.
			INT[1].1	1-버퍼가 활성화됩니다. 0-버퍼가 비활성화됩니다.
			INT[1].2	1-버퍼에 데이터가 저장되어 있습니다. 0-버퍼가 비어 있습니다.
			INT[1].3	1-버퍼가 꽉 찼습니다. 0-버퍼가 꽉 차지 않았습니다.
			INT[1].4	1-초기화 상태 메세지가 전송되지 않습니다(구독 시 및 중복 절체 시). 0-초기화 상태 메세지가 전송됩니다.

			기타 모든 비트는 예약되어 있으며 0 으로 설정됩니다.	
BufferSize	INT[2]	GSV	지정된 AlarmBuffer 객체의 버퍼 크기(kB)를 반환합니다. 개별 인스턴스의 버퍼 크기를 가져오려면 알람 버퍼 인스턴스 ID 를 지정해야 합니다.	
			INT[0].0	1-BufferSize 속성이 유효합니다. 0-BufferSize 속성이 유효하지 않습니다.
			INT[1]	BufferSize 속성 값.
BufferUsage	INT[2]	GSV	지정된 AlarmBuffer 객체에서 사용하는 버퍼 공간의 비율을 반환합니다. 개별 인스턴스의 버퍼 사용 값을 가져오려면 AlarmBuffer 인스턴스 ID 를 지정해야 합니다.	
			INT[0].1	1-BufferUsage 속성이 유효합니다. 0-BufferUsage 속성이 유효하지 않습니다.
			INT[1]	BufferUsage 속성 값.
SubscriberName	STRING	GSV	지정된 AlarmBuffer 객체의 구독자 이름을 반환합니다. 개별 인스턴스의 구독자 이름을 가져오려면 AlarmBuffer 인스턴스 ID 를 지정해야 합니다. 모든 문자열 유형이 대상 태그로 참조될 수 있습니다. 구독자 이름이 제공된 대상 태그 문자열에 맞지 않는 경우 명령어는 대상 태그에 맞을 수 있는 이름의 일부만 제공합니다. 명령어 호출 시 인스턴스 ID 가 지정한 AlarmBuffer 객체 인스턴스가 없으면 문자열 길이(.LEN 구성원)가 0 으로 설정됩니다. 구독자가 AlarmBuffer 객체를 생성할 때 구독자 이름을 제공하지 않으면 AlarmBuffer 객체에 대한 생성 서비스를 호출할 때 사용한 연결과 관련된 장치 일련 번호로 구독자 이름 속성이 설정됩니다.	

GSV 명령어 예

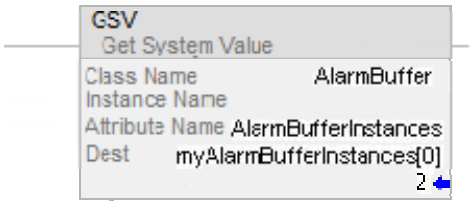
프로그램에는 컨트롤러의 현재 AlarmBufferInstances 목록을 가져오는 GSV 명령어가 포함되어 있을 수 있습니다. 이 명령어는 컨트롤러에 있는 각 AlarmBuffer 객체와 연관된 AlarmBuffer 객체 인스턴스 ID(DINT[1] – DINT[n-1])와 함께 현재 컨트롤러에 있는 알람 버퍼 객체(DINT[0])의 총 개수를 반환합니다. GSV 명령어는 Dest(대상) 태그 이름 아래에 AlarmBuffer 객체(DINT[0])의 개수 값을 표시합니다.

프로그램에서는 AlarmBuffer 객체 인스턴스 ID 를 사용하여 컨트롤러에 있는 AlarmBuffer 객체의 특정 인스턴스와 관련된 정보를 가져올 수 있습니다. 알람 버퍼 객체는 언제든지 생성 및

삭제될 수 있으므로 유효하거나 유효하지 않은 데이터를 나타내는 상태 워드(INT[0])는 AlarmBufferStatus, BufferSize 및 BufferUsage 속성의 대상 태그에 반환됩니다. 속성 이름이 AlarmBufferStatue, BufferSize 또는 BufferUsage 이면 반환되는 값은 (INT[1])입니다. 속성 이름이 SubscriberName 이면 반환되는 값은 구독자 이름입니다. SubsriberName 속성에 대한 상태는 반환되지 않습니다.

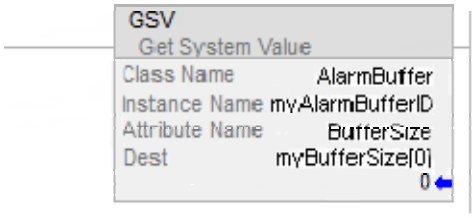
래더 다이어그램

다음은 AlarmBuffer 객체 ID 를 검색하는 GSV 명령어의 예입니다.



AlarmBufferInstances 의 GSV 가 값을 배열로 반환하더라도 배열 주소를 사용하여 해당 인스턴스에 대한 속성 값을 가져올 수는 없습니다. myAlarmBufferInstances[x](여기서, x = 1, 2, 3,...)의 값은 다음 설명에 나오는 myAlarmBufferID 처럼 직접(인덱싱되지 않은) 태그로 복사 또는 이동해야 합니다.

다음은 AlarmBuffer 객체의 버퍼 크기를 검색하는 GSV 명령어의 예입니다.



속성 이름이 AlarmBufferStatus, BufferSize 또는 BufferUsage 인 경우 Dest(대상) 태그 이름 아래에 표시되는 숫자는 유효하거나 유효하지 않은 비트 값입니다.

ST(스트럭처드 텍스트)

다음은 AlarmBuffer 객체 ID 를 검색하는 GSV 명령어의 예입니다.

- GSV(AlarmBuffer, AlarmBufferInstances, myAlarmBufferInstances[0]);

다음은 AlarmBuffer 객체를 검색하는 GSV 명령어의 예입니다.

- GSV(AlarmBuffer, myAlarmBufferID, BufferSize, myBufferSize[0]);

축 객체 액세스

AXIS 객체는 축에 관한 상태 정보를 제공합니다. 축 태그 이름을 지정하여 원하는 AXIS 객체를 결정합니다.

AXIS 객체에 대한 자세한 내용은 *SERCOS and Analog Motion Configuration and Startup User Manual*(발행 번호: MOTION-UM001)을 참조하십시오.

속성에 별표(*)가 표시되면 해당 속성이 ControlLogix 컨트롤러 및 모션 모듈 둘 다에 있다고 의미합니다. SSV 명령어를 사용하여 이러한 값 중 하나를 쓰면 컨트롤러가 모듈의 복사본을 자동으로 업데이트합니다. 그러나 이 프로세스는 즉시 처리되지는 않습니다. 축 상태 태그인 ConfigUpdateInProcess 가 제공되어 이 프로세스가 완료된 시점을 나타냅니다.

예를 들어 PositionLockTolerance 에 대해 SSV 를 수행한 경우 모듈이 성공적으로 업데이트될 때까지 축 태그의 ConfigUpdateInProcess 가 설정됩니다. 따라서 SSV 를 따르는 로직은 프로그램에서 계속 진행하기 전에 이러한 비트 리셋을 대기할 수 있습니다.

속성	데이터 유형	명령어	설명	
* AccelerationFeedForwardGain	REAL	GSV SSV	명령 가속도를 생성하는 데 필요한 토크 명령 출력 %.	
ACStopMode	SINT	GSV SSV	축에서 수행할 중지 유형.	
			값	의미
			0	빠른 중지
			1	빠른 종료
			2	하드 종료
ActualPosition	REAL	GSV	축의 위치 단위로 표시되는 실제 위치.	
ActualVelocity	REAL	GSV	위치 단위/초로 표시되는 축의 실제 속도.	
AnalogInput1	REAL	GSV SSV	이 속성은 Kinetix7000 드라이브인 아날로그 입력 2 와 연결된 축에만 적용됩니다. 정수 범위가 +/-16384 인 이 속성은 Kinetix7000 드라이브의 아날로그 입력에 연결된 아날로그 장치의 아날로그 값을 나타냅니다. 이러한 입력은 웹을 제어하는 드라이브에 직접 연결할 수 있는 로드셀(롤러에 대한 웹 힘 측정) 또는 댄서(웹 힘/위치 직접 측정)를 사용하는 웹/변환 응용 프로그램에 유용합니다.	

AverageVelocity	REAL	GSV	위치 단위/초로 표시되는 축의 평균 속도.		
AverageVelocityTimebase	REAL	GSV SSV	축 평균 속도의 시간 기준(초 단위).		
AxisConfigurationState	SINT	GSV	축 구성 상태.		
			값	의미	
			0 – 126	아직 구성되지 않음	
			127	사용된 축 데이터가 유효하지 않음(제작자와 소비자 간의 호환되지 않는 수정 버전이 원인임)	
			128	구성됨	
			3	응답 대기 중	
			4	구성됨	
AxisEventBits	DINT	GSV	서보 루프의 서보 이벤트 비트. (AXIS 구조에서 이 속성은 AxisEvent 구성원입니다.)		
			비트	비트 이름	의미
			0	WatchEventArmedStatus	아밍된 감시 이벤트
			1	WatchEventStatus	감시 이벤트
			2	RegEvent1ArmedStatus	아밍된 등록 이벤트
			3	RegEvent1Status	등록 이벤트
			4	HomeEventArmedStatus	아밍된 홈 이벤트
			5	HomeEventStatus	홈 이벤트
AxisState	SINT	GSV	축의 작동 상태.		
			값	의미	
			0	축이 준비됨	
			1	직접 드라이브 제어	
			2	서보 제어	
			3	축 폴트 발생	
			4	축 종료	
Bandwidth	REAL	GSV SSV	컨트롤러가 모션 적용 축 튜닝(MAAT) 명령어의 게인을 계산하는 데 사용하는 단위 게인 대역폭(Hz).		
C2CConnectionInstance	DINT	GSV	축 데이터를 생성하는 컨트롤러의 연결 인스턴스.		
C2CMapTableInstance	DINT	GSV	축 데이터를 생성하는 컨트롤러의 맵 인스턴스.		

CommandPosition	REAL	GSV	위치 단위로 표시되는 축의 명령 위치.
CommandVelocity	REAL	GSV	위치 단위로 표시되는 축의 명령 속도.
ConversionConstant	REAL	GSV SSV	단위를 피드백 카운트로 변환하는 데 사용되는 변환 계수(카운트/위치로 표시됨).
DampingFactor	REAL	GSV SSV	모션 실행 축 튜닝(MRAT) 명령어를 실행하는 중에 최대 위치 서보 대역폭을 계산하는 데 사용되는 값.
*DriveFaultAction	SINT	GSV SSV	드라이브 폴트가 발생할 때 수행되는 작업.

값	의미
0	축 종료
1	드라이브 비활성화
2	명령 모션 중지
3	상태 비트만 변경

DynamicsConfigurationBits	DINT	GSV SSV	수정 버전 16 에서는 컨트롤러가 S-곡선 프로파일에 대한 변경 사항을 처리하는 방법이 개선됩니다. 15 이전 수정 번전의 S-곡선의 동작으로 돌아가시겠습니까? NO - 이러한 비트를 켜둡니다(기본값). YES - 이러한 비트 중 하나 이상을 끕니다.

변경 사항 설정을 해제하려면	**비트 끄기**
S-곡선 중단 지연 감소 이 변경은 모션 축 정지(MAS) 명령어에 적용됩니다. 이를 통해 **가속** 축을 보다 빠르게 중지하기 위해 보다 높은 감속 저크를 사용할 수 있습니다. 현재 가속 저크보다 높은 경우 컨트롤러는 중지 명령어의 감속 저크를 사용합니다.	0

			S-곡선 속도 반전 감소 수정 버전 16 이전에서는 축을 감속하는 동안 감속 저크를 줄일 경우 축 방향이 일시적으로 바뀔 수 있습니다. 일반적으로 이는 조그를 재시작하거나 축이 중지된 동안 감속률을 낮추면서 이동하려고 하면 발생합니다. 이 변경 사항은 이러한 상황에서 축의 방향이 바뀌지 않도록 방지합니다.	1
			S-곡선 속도 오버슈트 감소 축을 가속하는 동안 가속 저크를 줄이면 축이 프로그래밍된 속도보다 오버슈트될 수 있습니다. 이 변경 사항은 프로그래밍된 속도의 50% 미만으로 오버슈트가 유지됩니다.	2
FaultConfigurationBits	DINT		축 유형	폴트 구성
*FeedbackFaultAction	SINT	GSV SSV	인코더 손실 폴트가 발생할 때 수행되는 작업.	
			값	의미
			0	축 종료
			1	드라이브 비활성화
			2	명령 모션 중지
			3	상태 비트만 변경
*FeedbackNoiseFaultAction	SINT	GSV SSV	인코더 노이즈 폴트가 발생할 때 수행되는 작업.	
			값	의미
			0	축 종료
			1	드라이브 비활성화
			2	명령 모션 중지
			3	상태 비트만 변경
*FrictionCompensation	REAL	GSV SSV	정지 마찰을 보상하기 위해 사용되는 고정된 출력 레벨(볼트 단위).	
GroupInstance	DINT	GSV	축을 포함한 모션 그룹의 인스턴스 번호.	

HardOvertravelFaultAction	SINT	GSV SSV	**값**	**의미**
			0	종료
			1	드라이브 비활성화
			2	모션 중지
			3	상태만
HomeConfigurationBits	DINT	GSV SSV	축의 모션 구성 비트.	
			비트	의미
			0	홈 방향
			1	홈 스위치 평상시 닫힘
			2	음의 홈 마커 에지
HomeMode	SINT	GSV SSV	축의 호밍 모드.	
			값	의미
			0	비활성 호밍
			1	능동 호밍(기본값)
			2	절대
HomePosition	REAL	GSV SSV	위치 단위로 표시되는 축의 호밍 위치.	
HomeReturnSpeed	REAL	GSV SSV	위치 단위/초로 표시되는 축의 호밍 반환 속도.	
HomeSequence	SINT	GSV SSV	축의 호밍 시퀀스 유형.	
			값	의미
			0	즉시 호밍
			1	스위치 호밍
			2	마커 호밍
			3	스위치-마커 호밍(기본값)
HomeSpeed	REAL	GSV SSV	위치 단위/초로 표시되는 축의 호밍 속도.	
Instance	DINT	GSV	축의 인스턴스 번호.	
InterpolatedActualPosition	REAL	GSV	시간 기준 위치 캡처의 경우 이 속성은 삽입된 실제 축 위치를 제공합니다. 위치는 위치 단위로 지정되며 InterpolationTime 속성 값에 따라 달라집니다. 실제 축 위치를 삽입하려면 SSV 명령어를 사용하여 InterpolationTime 속성을 설정합니다.	

<table>
<tr><td>InterpolatedCommandPosition</td><td>REAL</td><td>GSV</td><td>시간 기준 위치 캡처의 경우 이 속성은 삽입된 명령 축 위치를 제공합니다.
위치는 위치 단위로 지정되며 InterpolationTime 속성 값에 따라 달라집니다.
명령 축 위치를 삽입하려면 SSV 명령어를 사용하여 InterpolationTime 속성을 설정합니다.</td></tr>
<tr><td>InterpolationTime</td><td>DINT</td><td>GSV
SSV</td><td>시간 기준 위치 캡처에 대한 참조를 제공하려면 이 속성을 사용합니다.
위치를 삽입하려면 SSV 명령어를 사용하여 InterpolationTime 속성을 설정합니다. 그런 다음 컨트롤러에서는 다음 속성을 업데이트합니다.
• InterpolatedActualPosition
• InterpolatedCommandPosition
InterpolationTime 값을 제공하려면 다음과 같은 CST 타임스탬프를 생성하는 모든 이벤트를 사용할 수 있습니다.
• RegistrationTime 속성
• 디지털 출력 타임스탬프
InterpolationTime 속성은 CST 타임스탬프의 하위 32 비트만 사용합니다.</td></tr>
<tr><td>MapTableInstance</td><td>DINT</td><td>GSV</td><td>서보 모듈의 I/O 맵 인스턴스.</td></tr>
<tr><td>MasterOffset</td><td>REAL</td><td>GSV</td><td>위치 캠 마스터에 현재 적용된 위치 오프셋. 마스터 축의 위치 단위로 지정됩니다.</td></tr>
<tr><td>MaximumAcceleration</td><td>REAL</td><td>GSV
SSV</td><td>위치 단위/s^2로 표시되는 축의 최대 가속도.</td></tr>
<tr><td>MaximumDeceleration</td><td>REAL</td><td>GSV
SSV</td><td>위치 단위/s^2로 표시되는 축의 최대 감속도.</td></tr>
<tr><td>*MaximumNegativeTravel</td><td>REAL</td><td>GSV
SSV</td><td>위치 단위로 표시되는 음의 최대 트래블 제한.</td></tr>
<tr><td>*MaximumPositiveTravel</td><td>REAL</td><td>GSV
SSV</td><td>위치 단위로 표시되는 양의 최대 트래블 제한.</td></tr>
<tr><td>MaximumSpeed</td><td>REAL</td><td>GSV
SSV</td><td>위치 단위/초로 표시되는 축의 최대 속도.</td></tr>
<tr><td>ModuleChannel</td><td>SINT</td><td>GSV</td><td>서보 모듈의 채널.</td></tr>
<tr><td>MotionStatusBits</td><td>DINT</td><td>GSV</td><td>축의 모션 상태 비트. (AXIS 구조에서 이 속성은 MotionStatus 구성원입니다.)
<table>
<tr><td>비트</td><td>비트 이름</td><td>의미</td></tr>
<tr><td>0</td><td>AccelStatus</td><td>가속도</td></tr>
<tr><td>1</td><td>DecelStatus</td><td>감속도</td></tr>
<tr><td>2</td><td>MoveStatus</td><td>move</td></tr>
</table></td></tr>
</table>

			3	JogStatus	jog
			4	GearingStatus	gear
			5	HomingStatus	home
			6	StoppingStatus	정지
			7	AxisHomedStatus	호밍 상태
			8	PositionCamStatus	위치 캠
			9	TimeCamStatus	시간 캠
			10	PositionCamPendingStatus	보류 중인 위치 캠
			11	TimeCamPendingStatus	보류 중인 시간 캠
			12	GearingLockStatus	기어링 잠금
			13	PositionCamLockStatus	위치 캠 잠금
			14	MasterOffsetMoveStatus	마스터 오프셋 이동
			15	CoordinatedMotionStatus	좌표 모션
			16	TransformStateStatus	변환 상태
			17	ControlledByTransformStatus	변환으로 제어
*OutputLPFilterBandwidth	REAL	GSV SSV	서보 저역 디지털 출력 필터의 대역폭(Hz).		
*OutputLimit	REAL	GSV SSV	축의 최대 서보 출력 전압 값(볼트 단위).		
*OutputOffset	REAL	GSV SSV	서보 모듈 DAC 출력 및 서보 드라이브 입력의 누적 오프셋 영향을 오프셋하는 데 사용되는 값(볼트 단위).		
PositionError	REAL	GSV	축의 실제 위치와 명령 위치 간의 차이.		
*PositionErrorFaultAction	SINT	GSV SSV	위치 에러 폴트가 발생할 때 수행되는 작업.		
			값	의미	
			0	축 종료	
			1	드라이브 비활성화	
			2	명령 모션 중지	
			3	상태 비트만 변경	
*PositionErrorTolerance	REAL	GSV SSV	위치 에러 폴트가 발생하기 전에 서보가 허용하는 위치 단위로 표시되는 위치 에러의 양.		

PositionIntegratorError	REAL	GSV	위치 단위로 표시되는 축에 대한 위치 에러 합계.
*PositionIntegralGain	REAL	GSV SSV	정지 마찰 및 중력과 같은 외란에도 불구하고 정확한 축 위치를 얻기 위해 사용되는 값(1/ms^2).
PositionLockTolerance	REAL	GSV SSV	실제 위치 잠김 상태 표시를 제공할 때 서보 모듈이 허용하는 위치 단위로 표시되는 위치 에러의 양.
*PositionProportionalGain	REAL	GSV SSV	위치 에러를 수정하기 위해 컨트롤러가 위치 에러에 곱하는 값(1/밀리초).
PositionServoBandwidth	REAL	GSV SSV	컨트롤러가 모션 적용 축 튜닝(MAAT) 명령어의 게인을 계산하는 데 사용하는 단위 게인 대역폭.
*PositionUnwind	DINT	GSV SSV	회전 축의 자동 언와인드를 수행하는 데 사용되는 카운트 수/회전으로 표시되는 값.
ProcessStatus	INT	GSV	마지막 모션 실행 후크업 진단(MRHD) 명령어의 상태.

값	의미
0	테스트 프로세스에 성공함
1	테스트 진행 중
2	사용자가 테스트 프로세스를 중단함
3	테스트 시 2 초의 타임아웃을 초과함
4	서보 폴트로 인해 테스트 프로세스에 실패함
5	테스트 증가 부족

ProgrammedStopMode	SINT	GSV SSV	축에서 수행할 중지 유형.

값	의미
0	빠른 중지
1	빠른 종료
2	하드 종료

Registration1Position	REAL	GSV	위치 단위로 표시되는 축의 등록 위치.

<table>
<tr><td>RegistrationTime</td><td>DINT</td><td>GSV</td><td>이 속성을 사용하면 시간 기준 위치 캡처에 대한 타임스탬프를 제공할 수 있습니다.
RegistrationTime 속성에는 축 등록 이벤트의 CST 타임스탬프 중 하단 32 비트가 포함됩니다.
CST 타임스탬프는 마이크로초 단위로 측정됩니다.
축 등록 이벤트를 기반으로 위치를 삽입하려면:
GSV 명령어를 사용하여 RegistrationTime 속성의 값을 가져옵니다.
SSV 명령어를 사용하여 InterpolationTime 속성을 RegistrationTime 속성의 값으로 설정합니다.</td></tr>
<tr><td>RotaryAxis</td><td>SINT</td><td>GSV
태그</td><td>0 = 선형
1 = 회전
회전 축 속성이 참(1)으로 설정되면 축이 언와인드되도록 둡니다. 이 속성은 축이 전체 물리적 회전을 이동할 때마다 축의 위치를 언와인드하여 무한 위치 범위를 제공합니다. 축의 물리적 회전당 인코더 카운트의 수는 위치 언와인드 속성에 따라 지정됩니다. 직선 작업의 경우 카운트 수는 롤오버되지 않습니다. +/- 20 억으로 제한됩니다.</td></tr>
<tr><td>ServoFaultBits</td><td>DINT</td><td>GSV</td><td>서보 루프의 서보 폴트 비트. (AXIS 구조에서 이 속성은 AxisEvent 구성원입니다.)
<table>
<tr><th>비트</th><th>비트 이름</th><th>의미</th></tr>
<tr><td>0</td><td>PosSoftOvertravelFault</td><td>양의 오버트래블 폴트</td></tr>
<tr><td>1</td><td>NegSoftOvertravelFault</td><td>음의 오버트래블 폴트</td></tr>
<tr><td>2</td><td>PositionErrorFault</td><td>위치 에러 폴트</td></tr>
<tr><td>3</td><td>FeedbackFault</td><td>인코더 채널 A 손실 폴트</td></tr>
<tr><td>4</td><td>FeedbackFault</td><td>인코더 채널 B 손실 폴트</td></tr>
<tr><td>5</td><td>FeedbackFault</td><td>인코더 채널 Z 손실 폴트</td></tr>
<tr><td>6</td><td>FeedbackNoiseFault</td><td>인코더 노이즈 폴트</td></tr>
<tr><td>7</td><td>DriveFault</td><td>드라이브 폴트</td></tr>
<tr><td>8</td><td>ModuleSyncFault</td><td>동기식 연결 폴트</td></tr>
</table></td></tr>
</table>

			9	ModuleHardwareFault	서보 하드웨어 폴트
ServoOutputLevel	REAL	GSV	볼트로 표시되는 축 서보 루프에 대한 출력 전압 레벨.		
ServoStatusBits	DINT	GSV	서보 루프의 상태 비트. (AXIS 구조에서 이 속성은 ServoStatus 구성원입니다.)		
			비트	**비트 이름**	**의미**
			0	ServoActionStatus	서보 작업
			1	DriveEnableStatus	드라이브 활성화
			2	OutputLimitStatus	출력 제한
			3	PositionLockStatus	위치 잠금
			13	TuneStatus	튜닝 프로세스
			14	ProcessStatus	테스트 진단
			15	ShutdownStatus	축 종료
*SoftOvertravelFaultAction	SINT	GSV SSV	소프트 오버트래블 폴트가 발생할 때 수행되는 작업.		
			값	의미	
			0	축 종료	
			1	드라이브 비활성화	
			2	명령 모션 중지	
			3	상태 비트만 변경	
StartActualPosition	REAL	GSV	축에 대한 새 명령 모션이 시작되면 축의 위치 단위로 표시되는 실제 위치.		
StartCommandPosition	REAL	GSV	축에 대한 새 명령 모션이 시작되면 축의 위치 단위로 표시되는 명령 위치.		
StartMasterOffset	REAL	GSV	마지막 모션 축 이동(MAM) 명령어가 다음 이동 유형 중 하나를 실행할 때 마스터 오프셋. • AbsoluteMasterOffset • IncrementalMasterOffset 마스터 축의 위치 단위로 지정됩니다.		
StrobeActualPosition	REAL	GSV	모션 그룹 스트로브 위치(MGSP) 명령어가 실행될 때 축의 위치 단위로 표시되는 실제 위치.		
StrobeCommandPosition	REAL	GSV	모션 그룹 스트로브 위치(MGSP) 명령어가 실행될 때 축의 위치 단위로 표시되는 명령 위치.		

StrobeMasterOffset	REAL	GSV	모션 그룹 스트로브 위치(MGSP) 명령어가 실행될 때 마스터 오프셋. 마스터 축의 위치 단위로 지정됩니다.
TestDirectionForward	SINT	GSV	모션 실행 후크업 진단(MRHD) 명령어 실행 중 서보 모듈에 대해 축 트래블 방향.
			값 | 의미
			0 | 음의 방향(역방향)
			1 | 양의 방향(정방향)
TestIncrement	REAL	GSV SSV	모션 실행 후크업 진단(MRHD) 테스트를 시작하기 위해 필요한 모션의 양.
*TorqueScaling	REAL	GSV SSV	서보 루프의 출력을 드라이브에 등가 전압으로 변환하는 데 사용되는 값.
TuneAcceleration	REAL	GSV	마지막 모션 실행 축 튜닝(MRAT) 명령어 실행 중 위치 단위/s^2로 측정되는 가속도 값.
TuneAccelerationTime	REAL	GSV	마지막 모션 실행 축 튜닝(MRAT) 명령어 실행 중 초 단위로 측정되는 가속 시간.
TuneDeceleration	REAL	GSV	마지막 모션 실행 축 튜닝(MRAT) 명령어 실행 중 위치 단위/초로 측정되는 감속도 값.
TuneDecelerationTime	REAL	GSV	마지막 모션 실행 축 튜닝(MRAT) 명령어 실행 중 초 단위로 측정되는 감속 시간.
TuneInertia	REAL	GSV	마지막 모션 실행 축 튜닝(MRAT) 명령어 실행 중 컨트롤러의 측정값에서 계산된 축에 대한 mV/Kcounts/초로 표시되는 관성량 값.
TuneRiseTime	REAL	GSV	마지막 모션 실행 축 튜닝(MRAT) 명령어 실행 중 초 단위로 측정되는 축 상승 시간.
TuneSpeedScaling	REAL	GSV	마지막 모션 실행 축 튜닝(MRAT) 명령어 실행 중 mV/Kcounts/초 단위로 측정되는 축 드라이브 스케일링 계수.
TuneStatus	INT	GSV	마지막 모션 실행 축 튜닝(MRAT) 명령어의 상태.
			값 | 의미
			0 | 튜닝 프로세스에 성공함
			1 | 튜닝 진행 중
			2 | 사용자가 튜닝 프로세스를 중단함
			3 | 튜닝 시 2 초의 타임아웃을 초과함
			4 | 서보 폴트로 인해 튜닝 프로세스에 실패함

			5	축이 튜닝 트래블 제한에 도달함
			6	축 극성이 잘못 설정됨
			7	튜닝 속도가 너무 느려 측정할 수 없음
TuningConfigurationBits	DINT	GSV SSV	축의 튜닝 구성 비트.	
			비트	의미
			0	튜닝 방향(0=정방향, 1=역방향)
			1	튜닝 위치 에러 적분기
			2	튜닝 속도 에러 적분기
			3	튜닝 속도 피드포워드 비트
			4	가속도 피드포워드
			5	튜닝 속도 저역 필터
TuningSpeed	REAL	GSV SSV	마지막 모션 실행 축 튜닝(MRAT) 명령어로 시작된 위치 단위/초로 표시되는 최대 속도.	
TuningTravelLimit	REAL	GSV SSV	튜닝 중 작업을 제한하기 위해 모션 실행 축 튜닝(MRAT) 명령어에서 사용되는 트래블 제한.	
VelocityCommand	REAL	GSV	위치 단위/초로 표시되는 축의 속도 서보 루프에 대한 현재 속도 참조.	
VelocityError	REAL	GSV	위치 단위/초로 표시되는 서보 축의 명령 속도와 실제 속도 간의 차이.	
VelocityFeedback	REAL	GSV	위치 단위/초로 표시되는 서보 모듈에서 예측한 축의 실제 속도.	
*VelocityFeedforwardGain	REAL	GSV SSV	명령 속도를 생성하는 데 필요한 속도 명령 출력 %.	
*VelocityIntegralGain	REAL	GSV SSV	속도 에러를 수정하기 위해 컨트롤러가 VelocityIError 값에 곱하는 값(1/밀리초).	
VelocityIntegratorError	REAL	GSV	지정된 축에 대한 속도 에러.	
*VelocityProportionalGain	REAL	GSV SSV	속도 에러를 수정하기 위해 컨트롤러가 VelocityError 에 곱하는 값(1/밀리초).	
*VelocityScaling	REAL	GSV SSV	서보 루프의 출력을 드라이브에 등가 전압으로 변환하는 데 사용되는 값.	
VelocityServoBandwidth	REAL	GSV SSV	마지막 모션 실행 축 튜닝(MRAT) 명령어 실행 중 측정된 값에서 계산된 축에 대한 드라이브의 대역폭(Hz).	
WatchPosition	REAL	GSV	축의 위치 단위로 표시된 감시 위치.	

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

컨트롤러 객체 액세스

컨트롤러 객체는 컨트롤러 실행에 관한 상태 정보를 제공합니다.

속성	데이터 유형	명령어	설명
Audit Value	DINT[2], LINT	GSV	감사 값은 프로젝트가 컨트롤러에 다운로드되거나 이동식 저장소에서 로드되는 경우 생성되는 고유한 값입니다. 변경이 감지되면 이 값이 업데이트됩니다. 모니터링할 변경 사항을 지정하려면 ChangesToDetect 속성을 사용합니다. **팁:** Rockwell Automation 컨트롤러에서 LINT 데이터 유형을 사용하는 경우 DINT[2] 데이터 유형을 사용하여 제한 사항을 피할 것을 권장합니다.
ChangesToDetect	DINT[2], LINT	GSV, SSV	모니터링할 변경 사항을 지정하는 데 사용됩니다. 모니터링할 변경 사항이 발생하면 감사 값이 업데이트됩니다. **팁**: Rockwell Automation 컨트롤러에서 LINT 데이터 유형을 사용하는 경우 DINT[2] 데이터 유형을 사용하여 제한 사항을 피할 것을 권장합니다.
CanUseRPIFrom Producer	DINT	GSV	제작자가 지정한 RPI 를 사용할지 여부를 식별합니다. 값 의미 0 제작자가 지정한 RPI 사용 금지 1 제작자가 지정한 RPI 사용
ControllerLog Execution Modification Count	DINT	GSV SSV	프로그램/태스크 속성 변경, 온라인 편집 또는 컨트롤러 타임 슬라이스 변경으로 인해 생성된 컨트롤러 로그 항목 수. 또한 강제 적용으로 인해 생성된 로그 항목을 포함하도록 구성할 수도 있습니다. RAM 이 불량 상태가 되면 이 개수가 리셋됩니다. 이 개수는 가장 큰 DINT 에서 제한되지 않으므로 롤오버가 발생할 수 있습니다.
ControllerLog TotalEntryCount	DINT	GSV SSV	마지막 펌웨어 업그레이드 이후의 컨트롤러 로그 항목 수. RAM 이 불량 상태가 되면 이 개수가 리셋됩니다. 이 개수는 가장 큰 DINT 에서 제한됩니다.

DataTablePad Percentage	INT	GSV	별도로 확보해둔 사용 가능한 데이터 표 메모리의 비율(0 ~ 100).
IgnoreArrayFaultsDuringPost Scan	SINT	GSV SSV	SFC 동작을 사후 스캔할 때 발생하는 선택된 폴트를 억제하도록 구성하는 데 사용됩니다. SFC 가 자동 리셋으로 구성된 경우에만 유효합니다. • 0. 이 값은 사후 스캔 실행 중에 폴트를 억제하지 않습니다. 이 값이 기본값으로, 권장되는 동작입니다. • 1. 이 값은 SFC 작업을 사후 스캔하는 중 메이저 폴트 4/20(배열 첨자가 너무 큼) 및 4/83(값이 범위를 벗어남)을 자동으로 억제합니다. 폴트가 억제되면 컨트롤러는 내부 폴트 처리기를 사용해 해당 폴트를 자동으로 지웁니다. 그러면 폴트가 발생한 명령어는 자동으로 건너뛰고 다음 명령어에서 실행이 다시 시작됩니다. 폴트 처리기는 내장되어 있기 때문에 이러한 동작을 수행하도록 에러 처리기를 구성할 필요가 없습니다. 사실, 폴트 처리기를 구성하더라도 억제된 폴트가 에러 처리기를 트리거하지는 않습니다.
InhibitAutomatic FirmwareUpdate	BOOL	GSV SSV	펌웨어 슈퍼바이저를 활성화할지 여부를 식별합니다. • 0. 이 값은 펌웨어 슈퍼바이저를 실행합니다. • 1. 이 값은 펌웨어 슈퍼바이저를 실행하지 않습니다.
KeepTestEditsOnSwitch over	SINT	GSV	컨트롤러 절체 시 편십 태스트를 유지할지 여부를 식별합니다. • 0. 이 값은 절체에서 자동으로 편집 테스트를 취소합니다. • 1. 이 값은 절체에서 편집 테스트를 계속 수행합니다.
이름(Name)	문자열	GSV	컨트롤러 이름.
Redundancy 활성화됨	SINT	GSV	컨트롤러가 중복에 대해 구성되었는지 여부를 식별합니다. • 0. 이 값은 컨트롤러가 중복에 대해 구성되지 않았음을 나타냅니다. • 1. 이 값은 컨트롤러가 중복에 대해 구성되었음을 나타냅니다.

ShareUnused TimeSlice	INT	GSV SSV	연속 태스크 및 백그라운드 태스크가 사용되지 않는 타임 슬라이스 공유하는 방법을 식별합니다. • 0. 이 값은 백그라운드 태스크가 완료되었더라도 작업 시스템에서 연속 태스크에 대한 제어를 제공하지 않음을 나타냅니다. • 1. 이 값은 백그라운드 태스크가 완료되었더라도 연속 태스크가 실행됨을 나타냅니다. 이것이 기본값입니다. • 2. 이 값과 같거나 큰 값은 마이너 폴트를 로깅하고 설정은 변경하지 않고 그대로 둡니다.
TimeSlice	INT	GSV SSV	통신에 할당된 사용 가능한 CPU 의 비율(10 ~ 90). 키 스위치가 실행 위치에 있으면 이 값을 변경할 수 없습니다.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

ControllerDevice 객체 액세스

ControllerDevice 객체가 컨트롤러의 실제 하드웨어를 식별합니다.

속성	데이터 유형	명령어	설명
DeviceName	SINT[33]	GSV	컨트롤러와 메모리 보드의 카달로그 번호를 식별하는 ASCII 문자열. 첫 번째 바이트에는 배열 문자열에서 반환된 ASCII 문자 수가 포함되어 있습니다.
ProductCode	INT	GSV	각 값으로 컨트롤러 유형을 식별합니다. 15 SoftLogix5800 49 DriveLogix5725 포함 PowerFlex® 52 DriveLogix5730 포함 PowerFlex 53 에뮬레이터 54 1756-L61 ControlLogix 55 1756-L62 ControlLogix 56 1756-L63 ControlLogix 57 1756-L64 ControlLogix 64 1769-L31 CompactLogix 65 1769-L35E CompactLogix 67 1756-L61S GuardLogix 68 1756-L62S GuardLogix 69 1756-LSP GuardLogix

			72 1768-L43 CompactLogix 74 1768-L45 CompactLogix 76 1769-L32C CompactLogix 77 1769-L32E CompactLogix 80 1769-L35CR CompactLogix 85 1756-L65 ControlLogix 86 1756-L63S GuardLogix 87 1769-L23E-QB1 CompactLogix 88 1769-L23-QBFC1 CompactLogix 89 1769-L23E-QBFC1 CompactLogix 92 1756-L71 93 1756-L72 94 1756-L73 95 1756-L74 96 1756-L75 106 1769-L30ER 107 1769-L33ER 108 1769-L36ERM 109 1769-L30ER-NSE 110 1769-L33ERM 146 1756-L7SP 147 1756-L72S 148 1756-L73S 149 1769-L24ER-QB1B 150 1769-L24ER-QBFC1B 151 1769-L27ERM-QBFC1B 153 1769-L16ER-BB1B 154 1769-L18ER-BB1B 155 1769-L18ERM-BB1B 156 1769-L30ERM 158 1756-L71S
ProductRev	INT	GSV	현재 제품 버전을 식별합니다. 디스플레이는 16 진수여야 합니다. 하위 바이트는 주 수정 버전을, 상위 바이트는 부 수정 버전을 포함합니다.
SerialNumber	DINT	GSV	장치의 일련 번호. 장치를 제조할 때 일련 번호가 할당됩니다.
Status	INT	GSV	장치 상태 비트 7...4 의미 0000 예약됨 0001 플래시 업데이트 중 0010 예약됨 0011 예약됨 0100 플래시 불량 0101 폴트 모드 0110 실행

			0111 프로그램 폴트 상태 비트 11...8 의미 0001 복구 가능한 마이너 폴트 0010 복구 불가능한 마이너 폴트 0100 복구 가능한 메이저 폴트 1000 복구 불가능한 메이저 폴트 컨트롤러 상태 비트 13...12 의미 01 키스위치 실행 모드 10 키스위치 프로그램 모드 11 키스위치 원격 모드 15...14 의미 01 컨트롤러가 모드를 변경하는 중 10 디버깅 모드(컨트롤러가 실행 모드인 경우)
Type	INT	GSV	장치를 컨트롤러로 식별합니다. 컨트롤러 = 14.
Vendor	INT	GSV	장치의 공급업체를 식별합니다. Allen-Bradley = 0001.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

CoordiateSystem 객체 액세스

COORDINATESYSTEM 객체는 모션 좌표 시스템 실행에 관한 상태 정보를 제공합니다.

속성	데이터 유형	명령어	의미
CoordinateMotionStatus	DINT	GSV SSV	MCLM 또는 MCCM 명령어에 대해 축 잠금이 요청되고 축이 잠금 위치를 교차한 경우 설정됩니다. MCLM 또는 MCCM 이 시작되면 해제됩니다.
AccelStatus	BOOL	GSV SSV	벡터가 가속 중일 때 설정됩니다. 혼합이 진행 중이거나 벡터 이동이 일정한 속도 유지 또는 감속 중일 때 해제됩니다.
DecelStatus	BOOL	GSV SSV	벡터가 감속 중일 때 설정됩니다. 혼합이 진행 중이거나 벡터 이동이 가속 중 이동이 완료되면 해제됩니다.
ActualPosToleranceStatus	BOOL	GSV SSV	실제 허용 범위 종료 유형에 한해 설정됩니다. 다음 두 가지 조건이 충족된 후에 비트가 설정됩니다. 1) 보간이 완료됩니다. 2) 프로그래밍된 끝점까지 실제 거리가 구성된 좌표 시스템 실제 허용 범위 값보다 작습니다. 명령어가 완료된 후 설정된 채로 남아 있습니다. 새 명령어가 시작되면 리셋됩니다.
CommandPosToleranceStatus	BOOL	GSV SSV	프로그래밍된 끝점까지 거리가 구성 좌표 시스템의 명령 허용 범위 값보다 작을 때마다 모든 종료 유형에 대해 설정되고 명령어가 완료된 후 설정된 채로 남아 있습니다. 새 명령어가 시작되면 리셋됩니다.
StoppingStatus	BOOL	GSV SSV	MCCM 명령어가 실행될 때 Stopping Status 비트가 해제됩니다.
MoveStatus	BOOL	GSV SSV	MCCM 에 의해 축 모션이 시작될 때 설정됩니다. 마지막 모션 명령어 또는 정지를 유발하는 모션 명령어의 .PC 비트가 실행된 후 해제됩니다.
MoveTransitionStatus	BOOL	GSV SSV	감속 없음 또는 명령 허용 범위 종료 유형이 충족되면 설정됩니다. 혼합 공선이 이동할 경우 시스템이 항상 경로 위에 있으므로 비트가 설정되지 않습니다. 혼합이 완료되거나, 보류 중인 명령어 모션이 시작되거나, 정지를 발생시키는 모션 명령어가 실행될 때 해제됩니다. 경로 위에 있지 않음을 나타냅니다.

MovePendingStatus	BOOL	GSV SSV	이동 보류 중 비트는 조정 모션 명령어가 큐에 배치된 후 설정됩니다. 명령어가 실행되기 시작한 후 후속 조정 모션 명령어가 그 동안 큐에 배치되지 않는 한 비트가 해제됩니다. 조정 모션 명령어가 1 개인 경우 큐에서 실행으로 전환이 대략적 업데이트 기간보다 빠르게 일어나므로 Logix Designer 응용 프로그램의 사용자가 상태 비트를 감지하지 못할 수 있습니다. 명령어가 여러 개인 경우 비트 실제 값이 표시됩니다. 명령어가 명령어 큐에 있는 한 보류 중 비트가 설정됩니다. 이를 통해 Logix Designer 응용 프로그램 프로그래머는 복수의 조정 모션 명령어 실행을 간소화할 수 있는 수단을 얻게 됩니다. 프로그래머가 선행 명령어가 실행 중일 때 명령어가 큐에 배치되도록 허용할 경우 조정 모션 명령어가 포함된 래더 로직은 더 빠르게 실행될 수 있습니다. MovePendingStatus 비트가 해제되어 있으면 다음 조정 모션 명령어가 실행 가능합니다(즉, 큐 설정).
MovePendingQueueFullStatus	BOOL	GSV SSV	명령어 큐가 가득 찬 경우 설정됩니다. 새로운 조정 모션 명령어가 대기할 수 있는 여유가 있으면 해제됩니다.
TransformSourceStatus	BOOL	GSV SSV	좌표 시스템은 활성 변환의 소스입니다.
TransformTargetStatus	BOOL	GSV SSV	좌표 시스템은 활성 변환의 대상입니다.
CoorMotionLockStatus	BOOL	GSV SSV	MCLM 또는 MCCM 명령어에 대해 축 잠금이 요청되고 축이 잠금 위치를 교차한 경우 설정됩니다. MCLM 또는 MCCM 이 시작되면 해제됩니다. MCLM 또는 MCCM 명령어가 시작되자마자 즉시 정방향만 및 즉시 역방향만 열거형에 대한 비트가 설정됩니다. 마스터 축이 지정된 방향의 잠금 위치를 교차할 때 열거형이 위치 정방향만 또는 위치 역방향만인 경우 비트가 설정됩니다. 열거형이 없음인 경우 비트가 설정되지 않습니다. 마스터 축의 방향이 반전되거나 마스터 축을 따라 슬레이브 축이 정지하는 경우 CoordMotionLockStatus 비트가 해제됩니다. 마스터 축을 따라 슬레이브 좌표 시스템이 다시 시작되는 경우 CoordMotionLockStatus 비트가 다시 설정됩니다. MCS 명령어가 시작되어도 CoordMotionLockStatus 비트가 해제됩니다.

coordinateDefinition	UDINT	GSV	기하 구조의 좌표에 대한 좌표 정의
zeroAngleOffset4	REAL	GSV/SSV	데카르트 좌표가 아닌 기하 구조의 네 번째 축의 0 도 방향입니다.
zeroAngleOffset5	REAL	GSV/SSV	데카르트 좌표가 아닌 기하 구조의 다섯 번째 축의 0 도 방향입니다.
zeroAngleOffset6	REAL	GSV/SSV	데카르트 좌표가 아닌 기하 구조의 여섯 번째 축의 0 도 방향입니다.
linkLength3	REAL	GSV/SSV	로봇의 손목 링크의 선형 길이입니다.
ballScrewPitch	REAL	GSV/SSV	SCARA 독립 결합 나사의 피치입니다.
ActiveToolFrameID	DINT	GSV/태그	사용자가 MCTO 명령어에 지정한 활성 도구 식별자입니다.
MaxOrientationSpeed	REAL	GSV/SSV	좌표 시스템 방향 축의 최대 속도입니다.
MaxOrientationAccel	REAL	GSV/SSV	좌표 시스템 방향 축의 최대 가속도입니다.
MaxOrientationDecel	REAL	GSV/SSV	좌표 시스템 방향 축의 최대 감속도입니다.
ActiveWorkFrameID	REAL	GSV/태그	활성 작업 프레임
SwingArmOffsetA3	REAL	GSV/SSV	5 축 델타 기하 구조의 하단 베이스 플레이트의 중심에서 관절 4 프레임까지 X 축을 따라 적용되는 오프셋입니다.
SwingArmOffsetD3	REAL	GSV/SSV	5 축 델타 기하 구조의 하단 베이스 플레이트의 중심에서 관절 4 프레임까지 Z 축을 따라 적용되는 오프셋입니다.
SwingArmOffsetA4	REAL	GSV/SSV	5 축 델타 기하 구조의 관절 5 프레임까지 X 축 J4 프레임을 따라 적용되는 오프셋입니다.
SwingArmOffsetD4	REAL	GSV/SSV	5 축 델타 기하 구조의 관절 5 프레임까지 Z 축 J4 프레임을 따라 적용되는 오프셋입니다.
SwingArmOffsetD5	REAL	GSV/SSV	5 축 델타 기하 구조의 EOA 프레임까지 Z 축 J5 프레임을 따라 적용되는 오프셋입니다.
SwingArmCouplingRatioNum	UINT16	GSV/SSV	회전 축 대 틸트 축의 비율입니다.
SwingArmCouplingRatioDen	UINT16	GSV/SSV	회전 축 대 틸트 축의 비율입니다.
SwingArmCouplingDirection	UINT	GSV/SSV	델타 J1J2J3J4J5 로봇 기하 구조의 J5 틸트 축에 대한 연결된 J4 회전 축의 상대적인 방향입니다.

MotionGroup 객체 액세스

MOTIONGROUP 객체는 서보 모듈의 축 그룹에 대한 상태 정보를 제공합니다. 모션 그룹 태그 이름을 지정하여 원하는 MOTIONGROUP 객체를 정합니다.

속성	데이터 유형	명령어	설명
Alternate1UpdateMultiplier	USINT	GSV	축 업데이트 기간은 대체 1 업데이트 일정과 연결되어 있습니다.
Alternate1UpdatePeriod	UDINT	GSV	축 업데이트 기간은 대체 1 업데이트 일정과 연결되어 있습니다. 값은 대체 1 업데이트 기간 및 기본 업데이트 기간의 곱입니다.
Alternate2UpdateMultiplier	USINT	GSV	축 업데이트 기간은 대체 2 업데이트 일정과 연결되어 있습니다.
Alternate2UpdatePeriod	UDINT	GSV	축 업데이트 기간은 대체 2 업데이트 일정과 연결되어 있습니다. 값은 대체 1 업데이트 기간 및 기본 업데이트 기간의 곱입니다.
AutoTagUpdate	USINT	GSV SSV	모션 상태 속성의 자동 변환 및 업데이트를 제어합니다.
CoarseUpdatePeriod	UDINT	GSV	일반적으로 기본 업데이트 기간을 기본 업데이트 기간이라고 합니다.
Cycle Start Time	LTIME	GSV	64 비트 값(ms)은 업데이트 주기가 시작되는 타이머 이벤트에 해당합니다.
INSTANCE	DINT	GSV	이 MOTION_GROUP 객체의 인스턴스 수
MaximumInterval	LTIME	GSV SSV	이 태스크의 연속 실행 최대 범위
MinimumInterval	LTIME	GSV	이 태스크의 연속 실행 최소 범위
StartTime	LTIME	GSV	태스크를 마지막으로 실행했을 때 벽시계 시간 값
TaskAverageIOTime	UDINT	GSV SSV	평균 모션 태스크 입출력 시간은 모션 태스크 시작부터 연결 데이터 송신까지 경과된 시간입니다. (시간 상수 = 250 CUP)
TaskAverageScanTime	UDINT	GSV SSV	평균 모션 태스크 스캔 시간. (시간 상수 = 250 CUP)
TaskLastIOTime	UDINT	GSV	마지막 모션 태스크 입출력 시간은 모션 태스크 시작부터 연결 데이터 송신까지 경과된 시간입니다.
TaskLastScanTime	UDINT	GSV	마지막 모션 태스크 스캔 시간. (경과 시간)
TaskMaximumIOTime	UDINT	GSV SSV	최대 모션 태스크 입출력 시간은 모션 태스크 시작부터 연결 데이터 송신까지 경과된 시간입니다.
TaskMaximumScanTime	UDINT	GSV SSV	최대 모션 태스크 스캔 시간. (경과 시간)
Time Offset	LTIME	GSV	벽시계 시간 및 현재 주기 시작 시간 값에 연결된 컨트롤러의 로컬 타이머 값 사이에 시간 오프셋.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

메시지 객체 액세스

GSV/SSV 명령어를 통해 메시지 객체에 액세스할 수 있습니다. 메시지 태그 이름을 지정하여 원하는 메시지 객체를 결정합니다. 메시지 객체는 피투피 통신을 설정 및 트리거하는 인터페이스를 제공합니다. 이 객체는 PLC-5 프로세서의 MG 데이터 유형을 대체합니다.

속성	데이터 유형	명령어	설명
ConnectionPath	SINT[130]	GSV SSV	연결 경로를 설정하는 데이터. 처음 2 바이트(낮은 바이트 및 높은 바이트)는 연결 경로의 길이(바이트)입니다.
ConnectionRate	DINT	GSV SSV	연결의 요청된 패킷 속도.
MessageType	SINT	GSV SSV	메시지 유형을 지정합니다. 이 값에는 특정한 의미가 있습니다. • 0. 초기화되지 않음
포트	SINT	GSV SSV	메시지를 전송해야 하는 포트를 나타냅니다. 값별 의미: • 1. 백플레인 • 2. 시리얼 포트
Timeout Multiplier	SINT	GSV SSV	연결이 타임아웃되어 종료되었다고 고려해야 하는 시점을 결정합니다. 값별 의미: • 0. 연결이 업데이트 속도의 4 배로 타임아웃됩니다. 이것이 기본값입니다. • 1. 연결이 업데이트 속도의 8 배로 타임아웃됩니다. • 2. 연결이 업데이트 속도의 16 배로 타임아웃됩니다.
연결 안 됨 Timeout	DINT	GSV SSV	연결되지 않은 모든 메시지에 대한 타임아웃(마이크로초). 기본값은 30,000,000 마이크로초(30 s).

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

CST 객체 액세스

조정 시스템 시간(CST) 객체는 섀시 하나의 장치에 대해 조정 시스템 시간을 제공합니다.

속성	데이터 유형	명령어	설명
CurrentStatus	INT	GSV	조정 시스템 시간의 현재 상태. 각 비트에는 특정한 의미가 있습니다. • 0. 타이머 하드웨어에서 폴트 발생. 장치의 내부 타이머 하드웨어가 폴트 상태입니다. • 1. 램핑이 활성화됨. 타이머의 하위 16+ 비트의 현재 값이 더 낮은 값을 유지하는 대신 요청된 값으로 램핑됩니다. • 2. 시스템 시간 마스터. CST 객체는 ControlLogix 시스템의 마스터 시간 소스입니다. • 3. 동기화됨. CST 객체의 64 비트 CurrentValue 는 시스템 시간 업데이트를 통해 마스터 CST 객체에서 동기화됩니다. • 4. 로컬 네트워크 마스터. CST 객체는 로컬 네트워크 마스터 시간 소스입니다. • 5. 릴레이 모드. CST 객체는 시간 릴레이 모드에서 작동합니다. • 6. 중복 마스터가 감지됨. 중복 로컬 네트워크 시간 마스터가 감지됩니다. 시간 종속 노드의 경우 이 비트는 항상 0 입니다. • 7. 사용 안 함. • 8-9. 00. 시간 종속 노드 • 01. 시간 마스터 노드 • 10. 시간 릴레이 노드 • 11. 사용 안 함. • 10-15. 사용 안 함.
CurrentValue	DINT[2]	GSV	타이머의 현재 값. DINT[0]에는 하취 32 비트, DINT[1]에는 상위 32 비트가 포함되어 있습니다. 시간 소스는 업데이트 서비스와 로컬 통신 네트워크 동기화에서 제공되는 값과 일치하도록 조정됩니다. CurrentStatus 속성에 보고된 바와 같이 이 조정은 요청된 값에 대한 램핑이거나 요청 값에 대한 즉시 설정입니다.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

데이터로그 객체 액세스

DATALOG 객체는 특정 데이터 로그에 관한 상태 정보를 제공합니다. 데이터 로그 이름을 지정하여 원하는 DATALOG 객체를 결정합니다.

속성	데이터 유형	표준 태스크 내 명령어	안전 태스크 내 명령어	설명
CaptureFull	BOOL	GSV	없음	상태는 다음 둘 중 하나를 표시합니다. • 가장 최근의 데이터 캡처가 샘플 수집을 정지함, 또는 • 가장 최근의 데이터 캡처에서 가장 오래된 샘플이 캡처 크기의 초과로 인해 덮어 쓰여짐
CollectionCapacity	DINT	GSV	없음	각 컨트롤러 유형에 대해 초당 수집할 수 있는 바이트 수를 결정하는 컨트롤러 제공 주파수를 표시합니다. 데이터 로그에 사용되는 CPU 비율은 이 주파수 및 컨트롤러가 구성된 모든 데이터 로그에 대해 수집해야 하는 바이트 수에 기초하여 계산될 수 있습니다.
CollectionState	INT	GSV	없음	데이터 로그의 현재 데이터 수집 상태를 표시합니다. 데이터 수집 상태는 다음과 같습니다. • **오프라인** – 컨트롤러와 연결되어 있지 않습니다. • **비활성화** – 데이터 로그가 활성화될 때까지 데이터 로그 작업을 수행하지 않습니다. • **트리거 대기** – 시작 트리거 또는 스냅샷 트리거 대기 중입니다. 정지 트리거 대기 상태는 샘플 수집과 혼합 설정할 수 있습니다. 이 상태는 캡처 풀과 공존할 수 있습니다. • **샘플 수집** – 사전 샘플이나 사후 샘플이 아닌 샘플을 적극적으로 수집합니다. 사전 샘플링 수집 상태는 트리거 대기 상태와 혼합 설정할 수 있습니다. 이 상태는 캡처 풀과 공존할 수 있습니다. • **사후 샘플링 수집** – 정지 트리거가 발생하고 사후 샘플링을 수집합니다. 이

				상태는 캡처 풀과 공존할 수 있습니다. • **캡처 풀** – 캡처 크기를 초과했기 때문에 최근 데이터 캡처에서 샘플 수집을 중지하거나 최근 데이터 캡처의 가장 오래된 샘플을 덮어씁니다. 이 상태는 트리거 대기, 샘플 수집, 사후 샘플링 수집 또는 데이터 로그 풀과 공존할 수 있습니다. • **데이터 로그 풀** – 데이터 로그 작업이 데이터 캡처 초과로 인해 정지됩니다. 이 상태는 캡처 풀과 공존할 수 있습니다. 데이터 수집은 리셋 명령을 내리거나, 지우기 명령을 내린 후에 활성화 명령을 내리는 방식으로 다시 수행할 수 있습니다. • **폴트** - 폴트가 발생하면서 데이터 수집이 정지됩니다. 데이터 수집은 폴트가 지워지고 활성화 또는 리셋 명령을 내릴 때까지 더 이상 수행되지 않습니다. 이 상태는 캡처 풀과 공존할 수 있습니다.
CurrentCaptureNumber	INT	GSV	없음	현재의 캡처 수를 나타냅니다. 예를 들어, 구성값에서 유지할 데이터 캡처를 10 으로 설정한 경우, 현재의 캡처 수는 1 ~ 10 입니다.
DataCapturesToKeep	SINT	GSV	없음	지정된 데이터 로그에 대해 유지할 데이터 캡처의 구성된 수를 나타냅니다.
활성화됨	SINT	GSV	없음	지정된 데이터 로그가 활성화되는지 여부를 나타냅니다.
FaultReason	INT	GSV	없음	현재 발생한 폴트의 이유를 나타냅니다.
PreviousCaptureUsedStorage	DINT	GSV	없음	이전 데이터 캡처에서 사용된 저장 용량을 나타냅니다.
ReservedStorage	DINT	GSV	없음	현재의 데이터 로그를 위해 예약된 저장 용량이 총 저장 용량에서 차지하는 비율을 나타냅니다.
UsedStorage	DINT	GSV	없음	현재의 데이터 로그에서 수집된 데이터 샘플로 채워진 저장 용량이 차지하는 비율(%)을 나타냅니다.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

DF1 객체 액세스

DF1 객체는 시리얼 포트에 대해 구성할 수 있는 DF1 통신 드라이버에 대한 인터페이스를 제공합니다.

속성	데이터 유형	명령어	설명
ACKTimeout	DINT	GSV	메세지 전송 확인을 위해 대기하는 시간(점대점 및 마스터에만 해당). 유효한 값은 0 ~ 32,767 임. 20 밀리초 기간의 카운트 지연. 기본값은 50(1 초)임.
Diagnostic Counters	INT[19]	GSV	DF1 통신 드라이버에 대한 진단 카운터 배열.

워드 오프셋		DF1 점대점	DF1 스레이브 마스터
0	서명(0x0043)	서명(0x0042)	서명(0x0044)
1	모뎀 비트	모뎀 비트	모뎀 비트
2	패킷이 전송됨	패킷이 전송됨	패킷이 전송됨
3	패킷이 수신됨	패킷이 수신됨	패킷이 수신됨
4	패킷이 전달되지 않음	패킷이 전달되지 않음	패킷이 전달되지 않음
5	사용 안 함	메세지가 다시 시도됨	메세지가 다시 시도됨
6	NAK 가 수신됨	NAK 가 수신됨	사용 안 함
7	ENQ 가 수신됨	폴 패킷이 수신됨	사용 안 함
8	잘못된 패킷이 NAK 됨	잘못된 패킷이 ACK 되지 않음	잘못된 패킷이 ACK 되지 않음
9	NAK 를 전송한 메모리가 없음	ACK 되지 않은 메모리가 없음	사용 안 함
10	중복 패킷이 수신됨	중복 패킷이 수신됨	중복 패킷이 수신됨
11	잘못된 문자가 수신됨	사용 안 함	사용 안 함
12	DCD 복구 횟수	DCD 복구 횟수	DCD 복구 횟수
13	손실된 모뎀 수	손실된 모뎀 수	손실된 모뎀 수
14	사용 안 함	사용 안 함	최대 최우선 스캔 시간
15	사용 안 함	사용 안 함	마지막 최우선 스캔 시간
16	사용 안 함	사용 안 함	최대 정상 스캔 시간

17	사용 안 함	사용 안 함	마지막 정상 스캔 시간
18	ENQ 전송됨	사용 안 함	사용 안 함

Duplicate Detection	SINT	GSV	중복 메세지 감지가 활성화됨. 값별 의미: • 0. 중복 메세지 감지가 비활성화됨. • 0 이외의 값. 중복 메세지 감지가 활성화됨.
Embedded ResponseEnable	SINT	GSV	포함된 응답 기능을 활성화함(점대점에만 해당). 값별 의미: • 0. 하나가 수신된 후에만 시작됨. 이것이 기본값입니다. • 1. 무조건 활성화됨.
EnableStoreFwd	SINT	GSV	메세지를 수신하면 저장 및 전달 동작을 활성화함. 값별 의미: • 0. 메세지를 전달하지 않음 • 0 이외의 값. 메세지를 수신하면 저장 및 전달 표를 참조하십시오. 이것이 기본값입니다.
ENQTransmit Limit	SINT	GSV	ACK 타임아웃 후 전송할 조회(ENQ) 수(점대점에만 해당). 유효한 값은 0 ~ 127 임. 기본 설정은 3 임.
EOTSuppression	SINT	GSV	폴 패킷에 대한 응답으로 EOT 전송 억제를 활성화함(슬레이브에만 해당). 값별 의미: • 0. EOT 억제가 비활성화됨. • 0 이외의 값. EOT 억제가 활성화됨.
ErrorDetection	SINT	GSV	에러 감지 스키마를 지정함. 값별 의미: • 0. BCC. 이것이 기본값입니다. • 1. CRC.
MasterMessageTransmit	SINT	GSV	마스터 메세지 전송의 현재 값(마스터에만 해당). 값별 의미: • 0. 스테이션 폴 사이. 이것이 기본값입니다. • 1. 폴 시퀀스에서. 마스터의 스테이션 번호를 대체합니다.
MaxStation Address	SINT	GSV	DH-485 네트워크에 대한 최대 노드 주소의 현재 값(0 ~ 31). 기본값은 31 임.
NAKReceiveLimit	SINT	GSV	전송 중지 전에 메세지에 대한 응답으로 수신된 NAK 수(점대점 통신에만 해당). 유효한 값은 0 ~ 127 임. 기본값은 3 임.

NormalPollGroupSize	INT	GSV	최우선 폴링 노드 배열에서 모든 스테이션을 폴링한 후 정상 폴링 노드 배열에서 폴링할 스테이션 수 (마스터에만 해당). 유효한 값은 0 ~ 255 임. 기본값은 0 임.
PollingMode	SINT	GSV	현재 폴링 모드(마스터에만 해당). 기본 설정은 1 임. 값별 의미: • 0. 메세지 기반. 그러나 슬레이브가 메세지를 시작할 수는 없습니다. • 1. 메세지 기반. 그러나 슬레이브가 메세지를 시작할 수 있습니다. 이것이 기본값입니다. • 2. 표준. 노드 스캔마다 하나의 메세지를 전송함. • 3. 표준. 노드 스캔마다 다수의 메세지를 전송함.
ReplyMessage Wait	DINT	GSV	ACK 를 수신한 후 응답을 위해 슬레이브를 폴링하기 전 대기하는 (마스터로 작동하는) 시간(마스터에만 해당). 유효한 값은 0 ~ 65,535 임. 20 밀리초 기간의 카운트 지연. 기본값은 5 개 기간(100 밀리초)임.
SlavePollTimeout	DINT	GSV	마스터가 비활성화 상태이기 때문에 슬레이브가 전송할 수 없다고 선언하기 전에 마스터가 폴링하도록 슬레이브가 대기하는 시간(밀리초 단위). 유효한 값은 0 ~ 32,767 임. 20 밀리초 기간의 카운트 지연. 기본값은 3000 기간(1 분)임.
StationAddress	INT	GSV	시리얼 포트의 현재 스테이션 주소. 유효한 값은 0 ~ 254 임. 기본값은 0 임.
TokenHoldFactor	SINT	GSV	DH-485 네트워크에서 토큰을 전달하기 전에 이 노드에서 전송한 최대 메세지 수의 현재 값(1 ~ 4). 기본값은 1 임.
TransmitRetries	SINT	GSV	확인하지 않고 메세지를 다시 전송하는 횟수(마스터 및 슬레이브에만 해당). 유효한 값은 0 ~ 127 임. 기본값은 3 임.
PendingACK Timeout	DINT	SSV	ACKTimeout 속성에 대해 보류 중인 값.
Pending Duplicate Detection	SINT	SSV	DuplicateDetection 속성에 대해 보류 중인 값.
Pending Embedded ResponseEnable	SINT	SSV	EmbeddedResponse 속성에 대해 보류 중인 값.

PendingEnable StoreFwd	SINT	SSV	EnableStoreFwd 속성에 대해 보류 중인 값.
PendingENQ TransmitLimit	SINT	SSV	ENQTransmitLimit 속성에 대해 보류 중인 값.
PendingEOT Suppression	SINT	SSV	EOTSuppression 속성에 대해 보류 중인 값.
PendingError Detection	SINT	SSV	ErrorDetection 속성에 대해 보류 중인 값.
PendingMaster Message Transmit	SINT	SSV	MasterMessageTransmit 속성에 대해 보류 중인 값.
PendingMax StationAddress	SINT	SSV	MaxStationAddress 속성에 대해 보류 중인 값.
PendingNAK ReceiveLimit	SINT	SSV	NAKReceiveLimit 속성에 대해 보류 중인 값.
PendingNormal PollGroupSize	INT	SSV	NormalPollGroupSize 속성에 대해 보류 중인 값.
PendingPolling Mode	SINT	SSV	PollingMode 속성에 대해 보류 중인 값.
PendingReply MessageWait	DINT	SSV	ReplyMessageWait 속성에 대해 보류 중인 값.
PendingSlavePollTimeout	DINT	SSV	SlavePollTimeout 속성에 대해 보류 중인 값.
PendingStation Address	INT	SSV	StationAddress 속성에 대해 보류 중인 값.
PendingToken HoldFactory	SINT	SSV	TokenHoldFactor 속성에 대해 보류 중인 값.
PendingTransmitRetries	SINT	SSV	TransmitRetries 속성에 대해 보류 중인 값.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

FaultLog 객체 액세스

FaultLog 객체는 컨트롤러에 관한 폴트 정보를 제공합니다.

속성	데이터 유형	명령어	설명
MajorEvents	INT	GSV SSV	이 카운터가 리셋된 후 발생한 메이저 폴트 수.
MajorFaultBits	DINT	GSV SSV	개별 비트는 현재 메이저 폴트의 원인을 나타냅니다. 각 비트에는 특정한 의미가 있습니다. 1 전력 손실 3 I/O 4 명령어 실행(프로그램) 5 폴트 처리기 6 워치독 7 스택 8 모드 변경 11 모션
MinorEvents	INT	GSV SSV	이 카운터가 리셋된 후 발생한 마이너 폴트의 수.
MinorFaultBits	DINT	GSV SSV	개별 비트는 현재 마이너 폴트의 원인을 나타냅니다. 각 비트에는 특정한 의미가 있습니다. 4 - 명령어 실행(프로그램) 6 - 워치독 9 - 시리얼 포트 10 - 에너지 저장 모듈(ESM) 또는 무정전 전원 공급 장치(UPS) 20 - 라이센스/필수 CodeMeter 라이센스가 누락되었습니다.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

HardwareStatus 객체 액세스

HardwareStatus 객체는 CompactLogix 5480 컨트롤러 프로젝트에 대한 GSV 명령어를 사용하여 UPS, 팬 및 온도에 대한 상태 정보를 얻는 데 사용됩니다. 이 객체는 래더 다이어그램과 ST(스트럭처드 텍스트) 루틴 및 Add-On 명령어에서 지원됩니다.

속성	데이터 유형		명령어	설명
FanSpeeds	구조:		GSV	팬의 속도.
	팬 수	USINT		제품에서 지원하는 팬 수가 0 이면 장치가 팬을 지원하지 않습니다.
	팬 속도	2 팬의 경우 SINT[9]: SINT[0] = 팬 수 SINT[1-4] = 팬 1 속도 SINT[5-8] = 팬 2 속도		RPM
FanStatus	구조:		GSV	팬에 폴트가 발생했는지 여부를 나타냅니다.
	팬 상태 수 표시기	USINT		제품에서 지원하는 팬 수가 0 이면 장치가 팬 상태를 지원하지 않습니다.
	팬 상태	2 팬의 경우 SINT[3]: SINT[0] = 팬 수 SINT[1] = 팬 #1 상태 SINT[2] = 팬 #2 상태		• 0. 팬에 폴트가 없습니다. • 1. 팬에 폴트가 있습니다.
TemperatureFaultLevels	구조:		GSV	폴트 레벨(섭씨 온도)
	온도 폴트 레벨 수	USINT		온도 폴트 레벨 수가 0 이면 장치에서 온도 폴트 레벨을 지원하지 않습니다.
	온도 폴트 레벨	SINT[3] - 1 온도 센서: SINT[0] = 온도 폴트 레벨 수 SINT[1-2] = 온도 폴트 레벨 #1		온도(섭씨 온도)
온도	구조:		GSV	온도 값(섭씨 온도)
	온도 수	USINT		제품에서 지원하는 온도 수가 0 이면 장치가 온도를 지원하지 않습니다.

속성	데이터 유형		명령어	설명
	온도	SINT[3] - 1 온도 센서: SINT[0] = 온도 수 SINT[1-2] = 온도 #1		온도(섭씨 온도)
UPSBatteryFailure	SINT		GSV	UPS 배터리에 폴트가 발생했는지 여부를 나타냅니다. • 0. 연결된 UPS 배터리가 폴트를 감지하지 못했습니다. • 1. 연결된 UPS 가 연결된 배터리에 문제가 있음을 감지했습니다.
UPSBuffering	SINT		GSV	UPS 가 배터리를 통해 전원을 공급하는지 여부를 나타냅니다. • 0. UPS 가 배터리를 통해 전원을 공급하지 않습니다. • 1. UPS 가 배터리를 통해 전원을 공급합니다.
UPSInhibited	SINT		GSV	UPS 에 전원 분리를 요청합니다. • 0. 컨트롤러는 현재 전원을 제거하지 않으려고 합니다. • 1. UPS 가 전원 공급을 중지합니다.
UPSReady	SINT		GSV	충전이 85% 이상이고, 배선 폴트가 없고, 입력 전압이 충분하고, 금지 신호가 비활성 상태에서 UPS 가 준비되었는지 여부를 나타냅니다. • 0. UPS 가 준비되지 않음 • 1. UPS 가 준비됨
UPSSupported	SINT		GSV	UPS 가 지원되는지 여부를 나타냅니다. • 0. 지원되지 않음 • 1. 지원됨

메시지 객체 액세스

GSV/SSV 명령어를 통해 메시지 객체에 액세스할 수 있습니다. 메시지 태그 이름을 지정하여 원하는 메시지 객체를 결정합니다. 메시지 객체는 피투피 통신을 설정 및 트리거하는 인터페이스를 제공합니다. 이 객체는 PLC-5 프로세서의 MG 데이터 유형을 대체합니다.

속성	데이터 유형	명령어	설명
ConnectionPath	SINT[130]	GSV SSV	연결 경로를 설정하는 데이터. 처음 2 바이트(낮은 바이트 및 높은 바이트)는 연결 경로의 길이(바이트)입니다.
ConnectionRate	DINT	GSV SSV	연결의 요청된 패킷 속도.
MessageType	SINT	GSV SSV	메시지 유형을 지정합니다. 이 값에는 특정한 의미가 있습니다. • 0. 초기화되지 않음
포트	SINT	GSV SSV	메시지를 전송해야 하는 포트를 나타냅니다. 값별 의미: • 1. 백플레인 • 2. 시리얼 포트
Timeout Multiplier	SINT	GSV SSV	연결이 타임아웃되어 종료되었다고 고려해야 하는 시점을 결정합니다. 값별 의미: • 0. 연결이 업데이트 속도의 4 배로 타임아웃됩니다. 이것이 기본값입니다. • 1. 연결이 업데이트 속도의 8 배로 타임아웃됩니다. • 2. 연결이 업데이트 속도의 16 배로 타임아웃됩니다.
연결 안 됨 Timeout	DINT	GSV SSV	연결되지 않은 모든 메시지에 대한 타임아웃(마이크로초). 기본값은 30,000,000 마이크로초(30 s).

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

모듈 객체 액세스

모듈 객체는 모듈에 관한 상태 정보를 제공합니다. 특정 모듈 객체를 선택하려면 GSV/SSV 명령어의 객체 이름 피연산자를 모듈 이름으로 설정합니다. 지정한 모듈은 컨트롤러 관리자의 I/O 구성 섹션에 있어야 하며 장치 이름이 있어야 합니다.

속성	데이터 유형	명령어	설명
EntryStatus	INT	GSV	지정된 맵 항목의 현재 상태를 지정합니다. 비교 작업 수행 시 하위 12 비트가 마스킹되어야 합니다. 12 ~ 15 비트만 유효합니다. 값별 의미: • **16#0000. 대기.** 컨트롤러의 전원이 켜져 있습니다. • **16#1000. 폴트 발생.** 모듈 객체가 관련 모듈에 연결하지 못했습니다. 모듈에 연결하려고 할 때 모듈 객체가 주기적으로 이 상태를 벗어나므로 이 값을 모듈 실패를 확인하는 데 사용하면 안 됩니다. 대신 실행 상태(16#4000)를 테스트합니다. 모듈에서 폴트가 발생했는지 확인하기 위해 FaultCode 가 0 이 아닌지 확인합니다. 폴트가 발생한 경우 폴트 상태가 수정될 때까지 FaultCode 및 FaultInfo 속성이 유효합니다. • **16#2000. 유효성 확인.** 모듈 객체가 모듈에 연결하기 전에 모듈 객체 무결성을 확인하는 중입니다. • **16#3000. 연결 중.** 모듈 객체가 모듈에 대한 연결을 시작하고 있습니다. • **16#4000. 실행 중.** 모듈에 대한 모든 연결이 설정되었고 데이터가 전송되고 있습니다. • **16#5000. 종료 중.** 모듈 객체가 모듈에 대한 모든 연결을 종료하는 중입니다. • **16#6000. 금지됨.** 모듈 객체가 금지됩니다(Mode 속성에 금지 비트가 설정됨). • **16#7000. 대기 중.** 모듈 객체가 종속된 상위 객체가 실행 중이 아닙니다. • **16#9000 펌웨어 업데이트 중.** 펌웨어 슈퍼바이저가 모듈을 플래싱하려고 합니다. • **16#A000. 구성 중.** 컨트롤러가 모듈로 구성을 다운로드하고 있습니다.
FaultCode	INT	GSV	모듈 폴트(발생한 경우)를 식별하는 숫자.
FaultInfo	DINT	GSV	모듈 객체 폴트 코드에 대한 구체적인 정보를 제공합니다.
Firmware SupervisorStatus	INT	GSV	펌웨어 슈퍼바이저 기능의 현재 작동 상태를 식별합니다. 각 값에는 특정한 의미가 있습니다. • 0. 모듈 업데이트가 실행 중이 아닙니다. • 1. 모듈 업데이트가 실행 중입니다.

ForceStatus	INT	GSV	강제 적용 상태를 지정합니다. 각 비트에는 특정한 의미가 있습니다. • 0. 강제 적용이 설치됨(1 = 예, 0 = 아니요) • 1. 강제 적용이 활성화됨(1 = 예, 0 = 아니요)
Instance	DINT	GSV	이 모듈 객체의 인스턴스 번호를 제공합니다.
LEDStatus	INT	GSV	컨트롤러 전면에 있는 I/O 상태 표시기의 현재 상태를 지정합니다.(1) 각 값에는 특정한 의미가 있습니다. • 0. 상태 표시기 꺼짐: 컨트롤러에 대해 모듈 객체가 구성되어 있지 않습니다. (컨트롤러 관리자의 I/O 구성 섹션에 모듈이 없습니다.) • 1. 빨간색으로 깜박임: 실행 중인 모듈 객체가 없습니다. • 2. 녹색으로 깜박임: 하나 이상의 모듈 객체가 실행되고 있지 않습니다. • 3. 녹색: 모든 모듈 객체가 실행 중입니다. 이 속성이 전체 모듈 컬렉션에 적용되기 때문에 이 속성에서 객체 이름을 입력하지 않습니다.
Mode	INT	GSV SSV	모듈 객체의 현재 모드를 지정합니다. 각 비트에는 특정한 의미가 있습니다. • 0. 설정된 경우, 컨트롤러가 실행 모드일 때 모든 모듈 객체 연결 폴트가 있으면 메이저 폴트가 발생합니다. • 2. 설정된 경우 모듈에 대한 모든 연결이 종료된 후 모듈 객체가 금지된 상태를 입력하도록 합니다.
Path	SINT 배열	GSV	경로를 참조 중인 모듈로 지정합니다. 버전 24 소프트웨어에서 시작하는 새로운 속성입니다. 각 바이트에는 특정 의미가 있습니다. • 0-1. 경로 길이(바이트). 0 일 경우, SINT 배열 길이가 반환된 모듈 경로를 저장할 만큼 충분하지 않습니다. SINT 배열 길이가 경로를 저장할 만큼 충분하지 않은 경우 배열 값이 영이 되고 마이너 폴트가 기록됩니다.

(1) 1756-L7x 컨트롤러에는 컨트롤러 전면의 상태 표시기 표시가 없지만 이 기능을 사용합니다.

추가 참조

모듈 폴트: 16#0000 - 16#00ff 페이지의 300

모듈 폴트: 16#0100 - 16#01ff 페이지의 302

모듈 폴트: 16#0200 - 16#02ff 페이지의 308

모듈 폴트: 16#0300 - 16#03ff 페이지의 309

모듈 폴트: 16#0800 - 16#08ff 페이지의 312

모듈 폴트: 16#fd00 - 16#fdff 페이지의 313

모듈 폴트: 16#fe00 - 16#feff 페이지의 315

모듈 폴트: 16#ff00 - 16#ffff 페이지의 317

루틴 객체 액세스

루틴 객체는 루틴에 대한 상태 정보를 제공합니다. 루틴 이름을 지정하여 원하는 루틴 객체를 정합니다.

속성	데이터 유형	표준 태스크 내 명령어	안전 태스크 내 명령어	설명
Instance	DINT	GSV	GSV	루틴 객체의 인스턴스 번호 제공함. 유효값이 0 에서 65,535 입니다.
Name	문자열	GSV	GSV	루틴 이름.
SFCPaused	INT	GSV	없음	SFC 루틴에서 SFC 의 일시 중지 여부를 나타냅니다. 값별 의미: • 0. SFC 가 중지되지 않았습니다. • 1. SFC 가 중지됩니다.
SFCResuming	INT	GSV SSV	없음	SFC 루틴에서 SFC 가 실행을 다시 시작하는지 여부를 나타냅니다. 값별 의미: • 0. SFC 가 실행되고 있지 않습니다. 이 속성은 차트가 실행될 때 스캔 마지막에서 0 으로 자동 설정됩니다. • 1. SFC 가 실행되고 있습니다. 그렇게 구성된 경우 단계 및 액션 타이머가 이전 값을 보유합니다. 이 속성은 차트가 일시 중지 상태를 벗어난 후 첫 번째 스캔에서 1 로 자동 설정됩니다.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

중복 객체 액세스

REDUNDANCY 객체는 중복 시스템에 대한 상태 정보를 제공합니다.

다음 정보에 대해	다음 속성 가져오기	데이터 유형	GSV/SSV	설명	
전체 섀시의 중복 상태	ChassisRedundancyState	INT	GSV	**다음의 경우**	**그러면**
				16#2	기본 장치(동기화된 보조 장치 포함)
				16#3	기본 장치(자격이 부여되지 않은 보조 장치 포함)
				16#4	기본 장치(보조 장치 없음)
				16#10	업데이트하도록 잠겨 있는 기본 장치
파트너 섀시의 중복 상태	PartnerChassisRedundancyState	INT	GSV	**다음의 경우**	**그러면**
				16#8	동기화된 기본 장치
				16#9	자격이 부여되지 않은 보조 장치(기본 장치 포함)
				16#E	파트너 없음
				16#12	업데이트하도록 잠겨 있는 보조 장치
컨트롤러의 중복 상태	ModuleRedundancyState	INT	GSV	**다음의 경우**	**그러면**
				16#2	기본 장치(동기화된 보조 장치 포함)
				16#3	기본 장치(자격이 부여되지 않은 보조 장치 포함)
				16#4	기본 장치(보조 장치 없음)
				16#6	기본 장치(동기화 보조 장치 포함)
				16#F	업데이트하도록 잠금을 제공하는 기본 장치
				16#10	업데이트하도록 잠겨 있는 기본 장치

파트너의 중복 상태	PartnerModule RedundancyState	INT	GSV	**다음의 경우**	**그러면**
				16#7	동기화 중인 보조 장치
				16#8	동기화된 기본 장치
				16#9	자격이 부여되지 않은 보조 장치(기본 장치 포함)
				16#E	파트너 없음
				16#11	업데이트하도록 잠금 기능을 제공하는 보조 장치
				16#12	업데이트하도록 잠겨 있는 보조 장치
파트너 컨트롤러와 호환성 점검 결과	CompatibilityResults	INT	GSV	**다음의 경우**	**그러면**
				0	결정되지 않음
				1	호환되는 파트너가 없음
				2	전체 호환 가능한 파트너
동기화(자격 부여) 진행 상태	Qualification InProgress	INT	GSV	**다음의 경우**	**그러면**
				-1	동기화(자격 부여)가 진행 중이 아닙니다.
				0	지원되지 않음
				1...999	완료율을 측정할 수 있는 모듈의 경우, 완료된 동기화(자격 부여) 비율입니다.
				50	완료율을 측정할 수 없는 모듈의 경우, 완료된 동기화(자격 부여)이 진행 중입니다.
				100	동기화(자격 부여) 완료
컨트롤러와 파트너의 키 스위치 설정이 일치하거나 일치하지 않습니다.	KeyswitchAlarm	DINT	GSV	**다음의 경우**	**그러면**
				0	다음 중 하나가 참입니다. 키 스위치가 일치합니다. 파트너가 없습니다.
				1	키 스위치가 일치하지 않습니다.

파트너의 키 스위치 위치	PartnerKeyswitch	DINT	GSV	**다음의 경우**	**그러면**
				0	알 수 없음
				1	RUN
				2	PROG
				3	REM
파트너의 마이너 폴트 상태(ModuleRedundancyState 가 파트너가 있음을 나타낼 경우)	PartnerMinorFaults	DINT	GSV	**해당 비트**	**마이너 폴트를 나타냄**
				1	전원 켜기 폴트
				3	I/O 폴트
				4	명령어(프로그램) 오류
				6	주기 태스크 오버랩(워치독)
				9	시리얼 포트 오류(1756-L7x 프로젝트에서 사용 불가)
				10	저전력 또는 에너지 저장 모듈 오류
파트너 모드	PartnerMode	DINT	GSV	**다음의 경우**	**그러면**
				16#0	전원 켜기
				16#1	Program
				16#2	실행
				16#3	테스트(Test)
				16#4	폴트
				16#5	실행-프로그램
				16#6	테스트-프로그램
				16#7	프로그램-실행
				16#8	테스트-실행
				16#9	실행-테스트
				16#A	프로그램-테스트
				16#B	폴트 상태
				16#C	폴트-프로그램
한 쌍의 중복 섀시에서 섀시 상태를 고려하지 않고 특정 섀시 식별	PhysicalChassisID	INT	GSV	**다음의 경우**	**그러면**
				0	알 수 없음
				1	섀시 A
				2	섀시 B

섀시에 있는 Redundancy 모듈의 슬롯 번호(예를 들어 1756-RM, 1756-RM2)	SRMSlotNumber	INT	GSV		
마지막 크로스로드 크기 보조 섀시가 있는 경우 마지막 크로스로드 크기	LastDataTransferSize	DINT	GSV	이 속성은 마지막 스캔에서 크로스로드되거나 크로스로드될 수 있는 데이터 크기를 부여합니다. DINT 크기(4 바이트 단어). 중복에 대해 컨트롤러를 구성해야 합니다. 보조 섀시가 필요하지 않습니다. 동기화된 보조 섀시가 있습니까?	
				예	이렇게 하면 마지막 스캔에서 크로스로드된 DINT 수가 부여됩니다.
				아니요	이렇게 하면 마지막 스캔에서 크로스로드될지 못 한 DINT 수가 부여됩니다.
가장 큰 크로스로드 크기 보조 섀시가 있는 경우 가장 큰 크로스로드 크기	MaxDataTransferSize	DINT	GSV SSV	DINT 크기(4 바이트 단어). 중복에 대해 컨트롤러를 구성해야 합니다. 보조 섀시가 필요하지 않습니다. 값을 다시 설정하려면 소스 값이 0 인 SSV 명령어를 사용합니다. 동기화된 보조 섀시가 있습니까?	
				예	이렇게 하면 크로스로드된 가장 큰 DINT 수가 부여됩니다.
				아니요	이렇게 하면 크로스로드될지 못 한 가장 큰 DINT 수가 부여됩니다.

<table>
<tr><td rowspan="13">파트너 모드</td><td rowspan="13">PartnerMode</td><td rowspan="13">DINT</td><td rowspan="13">GSV</td><td>다음의 경우</td><td>그러면</td></tr>
<tr><td>16#0</td><td>전원 켜기</td></tr>
<tr><td>16#1</td><td>Program</td></tr>
<tr><td>16#2</td><td>실행</td></tr>
<tr><td>16#3</td><td>테스트(Test)</td></tr>
<tr><td>16#4</td><td>폴트</td></tr>
<tr><td>16#5</td><td>실행-프로그램</td></tr>
<tr><td>16#6</td><td>테스트-프로그램</td></tr>
<tr><td>16#7</td><td>프로그램-실행</td></tr>
<tr><td>16#8</td><td>테스트-실행</td></tr>
<tr><td>16#9</td><td>실행-테스트</td></tr>
<tr><td>16#A</td><td>프로그램-테스트</td></tr>
<tr><td>16#B</td><td>폴트 상태</td></tr>
<tr><td>16#C</td><td>폴트-프로그램</td></tr>
<tr><td rowspan="4">한 쌍의 중복 섀시에서 섀시 상태를 고려하지 않고 특정 섀시 식별</td><td rowspan="4">PhysicalChassisID</td><td rowspan="4">INT</td><td rowspan="4">GSV</td><td>다음의 경우</td><td>그러면</td></tr>
<tr><td>0</td><td>알 수 없음</td></tr>
<tr><td>1</td><td>섀시 A</td></tr>
<tr><td>2</td><td>섀시 B</td></tr>
<tr><td>섀시의 1757-SRM 모듈 슬롯 번호</td><td>SRMSlotNumber</td><td>INT</td><td>GSV</td><td colspan="2"></td></tr>
<tr><td rowspan="4">• 마지막 크로스로드 크기
• 보조 섀시가 있는 경우 마지막 크로스로드 크기</td><td rowspan="4">LastDataTransferSize</td><td rowspan="4">DINT</td><td rowspan="4">GSV</td><td colspan="2">이 속성은 마지막 스캔에서 크로스로드되거나 크로스로드될 수 있는 데이터 크기를 부여합니다.
• DINT 크기(4 바이트 단어).
• 중복에 대해 컨트롤러를 구성해야 합니다.
• 보조 섀시가 필요하지 않습니다.
동기화된 보조 섀시가 있습니까?</td></tr>
<tr><td>예</td><td>이렇게 하면 마지막 스캔에서 크로스로드된 DINT 수가 부여됩니다.</td></tr>
<tr><td>아니요</td><td>이렇게 하면 마지막 스캔에서 크로스로드될지 못 한 DINT 수가 부여됩니다.</td></tr>
</table>

• 가장 큰 크로스로드 크기 • 보조 섀시가 있는 경우 가장 큰 크로스로드 크기	MaxDataTransferSize	DINT	GSV SSV	이 속성은 LastDataTransfer 크기 속성에 가장 큰 수를 부여합니다. • DINT 크기(4 바이트 단어). • 중복에 대해 컨트롤러를 구성해야 합니다. • 보조 섀시가 필요하지 않습니다. • 값을 다시 설정하려면 소스 값이 0 인 SSV 명령어를 사용합니다. 동기화된 보조 섀시가 있습니까?	
				예	이렇게 하면 크로스로드된 가장 큰 DINT 수가 부여됩니다.
				아니요	이렇게 하면 크로스로드될지 못 한 가장 큰 DINT 수가 부여됩니다.

프로그램 객체 액세스

프로그램 객체는 프로그램에 대한 상태 정보를 제공합니다. 프로그램 이름을 지정하여 원하는 프로그램 객체를 정합니다.

속성	데이터 유형	표준 태스크 내 명령어	안전 태스크 내 명령어	설명
DisableFlag	SINT	GSV SSV	없음	프로그램의 실행을 제어함. 값별 의미: • 0. 실행이 활성화됨. • 0 이외의 값. 실행이 비활성화됨.
	DINT	GSV	GSV	0 이 아닌 값이 비활성화됩니다.
LastScanTime	DINT	GSV SSV	없음	마지막으로 실행되었을 때 프로그램 실행 시간. 시간은 마이크로초 단위입니다.
MajorFault Record	DINT[11]	GSV SSV	GSV SSV	이 프로그램의 메이저 폴트 기록함.

팁: MajorFaultRecord 속성에 간편하게 액세스할 수 있도록 사용자 정의 구조를 만드는 것이 좋습니다.

이름	데이터 유형	스타일	설명	
TimeLow	DINT	10 진수	폴트 타임스탬프 값의 하위 32 비트	
TimeHigh	DINT	10 진수	폴트 타임스탬프 값의 상위 32 비트	
유형	INT	10 진수	폴트 유형(프로그램, I/O 등)	
Code	INT	10 진수	폴트에 대한 고유 코드(폴트 유형에 따라 다름)	
Info	DINT[8]	16 진수	폴트 특정 정보(폴트 유형 및 코드에 따라 다름)	
MaxScanTime	DINT	GSV SSV	없음	이 프로그램에 대해 최대로 기록된 실행 시간. 시간은 마이크로초 단위입니다.
이름(Name)	문자열	GSV	GSV	프로그램 이름.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

안전 객체 액세스

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

안전 컨트롤러 객체는 안전 상태 및 안전 서명 정보를 제공합니다. SafetyTask 및 SafetyFaultRecord 속성으로 복구할 수 없는 폴트에 관한 정보를 캡처할 수 있습니다.

GuardLogix 컨트롤러 사용 설명서 (발행 번호 1756-UM020)을 참조하십시오.

속성	데이터 유형	표준 태스크 내 명령어	안전 태스크 내 명령어	설명
SafetyLockedState	SINT	GSV	없음	컨트롤러가 안전 잠금 상태인지 또는 잠금 해제되었는지 나타냅니다.
SafetySILConfiguration	SINT	GSV	없음	안전 SIL 구성을 지정합니다. • 2 -- SIL2/PLd • 3 -- SIL3/PLe

SafetyStatus	INT	GSV	없음	안전 상태를 지정합니다. 값별 의미: : • 1000000000000000 -- 안전 태스크 정상. • 1000000000000001 -- 안전 태스크 작동 불능. • 00000000000000000 -- 파트너 누락. • 00000000000000001 -- 파트너 사용할 수 없음. • 00000000000000010 -- 하드웨어 호환 안 됨. • 00000000000000011 -- 펌웨어 호환 안 됨.
SafetySignature Exists	SINT	GSV	GSV	안전 태스크 서명이 있는지 나타냅니다.
SafetySignature ID (Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러에만 적용 가능)	SINT	GSV	없음	32 비트 ID 번호.
SafetySignature (Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러에만 적용 가능)	문자열	GSV	없음	32 비트 ID 번호에는 ID 번호에 추가로 날짜 및 타임스탬프가 포함됩니다.
SafetyTaskFault Record	DINT[11]	GSV	없음	안전 태스크 폴트를 기록합니다.
SafetySignatureIDLong (Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용 가능)	SINT[33]	GSV	없음	바이트 배열의 32 바이트 안전 서명 ID. 첫 번째 바이트는 바이트 단위 안전 서명 ID 의 크기이고 남은 31 바이트는 서명 ID 입니다.
SafetySignatureIDHex (Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용 가능).	문자열	GSV	없음	서명 ID 의 64 자 16 진수 문자열 표시.
SafetySignatureDateTime (Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용 가능).	문자열	GSV	없음	안전 서명의 27 자 날짜 시간(mm/dd/yyyy, hh:mm:ss.iii<AM 또는 PM> 형식)

SerialPort 객체 액세스

SerialPort 객체는 시리얼 통신 포트에 대해 인터페이스를 제공합니다.

속성	데이터 유형	명령어	설명
BaudRate	DINT	GSV	전송 속도를 지정합니다. 유효 값은 110, 300, 600, 1200, 2400, 4800, 9600, 19200(기본값)입니다.
ComDriverID	SINT	GSV	특정 드라이버를 지정합니다. 값별 의미: • 0xA2. DF1. 이것이 기본값입니다. • 0xA3. ASCII.
DataBits	SINT	GSV	문자 당 데이터 비트 수를 지정합니다. 값별 의미: • 7. 데이터 비트가 7 개. ASCII 만 해당됩니다. • 8. 데이터 비트가 8 개. 이것이 기본값입니다.
DCDDelay	INT	GSV	패킷 에러를 반환하기 전에 데이터 캐리어 검색(DCD)이 낮아질 때까지의 대기 시간을 지정합니다. 지연이 초당 패킷 수로 표시합니다. 기본값이 0 카운터입니다.
Parity	SINT	GSV	패리티를 지정합니다. 값별 의미: • 0. 패리티가 없습니다. 이것이 기본값입니다. • 1. 홀수 패리티. ASCII 만 해당됩니다. • 2. 짝수 패리티.
RTSOffDelay	INT	GSV	마지막 문자가 전송된 후 RTS 라인 해제를 지연시킬 시간. 유효값: 0 ~ 32,767 20 밀리초 기간의 카운트 지연. 기본값은 0 밀리초입니다.
RTSSendDelay	INT	GSV	RTS 라인을 설정한 후 메시지의 첫 번째 문자 전송을 지연할 시간. 유효값: 0 ~ 32,767 20 밀리초 기간의 카운트 지연. 기본값은 0 밀리초입니다.
StopBits	SINT	GSV	스탑 비트 수를 지정합니다. 값별 의미: • 1. 스탑 비트가 1 개. 이것이 기본값입니다. • 2. 스탑 비트가 2 개. ASCII 만 해당됩니다.
PendingBaudRate	DINT	SSV	BaudRate 속성에 대한 보류 값.
PendingCOM DriverID	SINT	SSV	COMDriverID 속성에 대한 보류 값.
PendingDataBits	SINT	SSV	DataBits 속성에 대한 보류 값.
PendingDCD 지연	INT	SSV	DCDDelay 속성에 대한 보류 값.
PendingParity	SINT	SSV	Parity 속성에 대한 보류 값.

PendingRTSOff Delay	INT	SSV	RTSOffDelay 속성에 대한 보류 값.
PendingRTSSendDelay	INT	SSV	RTSSendDelay 속성에 대한 보류 값.
PendingStopBits	SINT	SSV	StopBits 속성에 대한 보류 값.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

태스크 객체 액세스

TASK 객체는 테스크에 대한 상태 정보를 제공합니다. 테스크 이름을 지정하여 원하는 TASK 객체를 정합니다.

속성	데이터 유형	표준 태스크 내 명령어	안전 태스크 내 명령어	설명
DisableUpdateOutputs	DINT	GSV SSV	없음	태스크 종료 시 출력 처리 활성화 또는 비활성화함. • 이 속성을 0 으로 설정하면 태스크 종료 시 출력 처리가 활성화됩니다. • 이 속성을 1(또는 0 이 아닌 모든 값)로 설정하면 태스크 종료 시 출력 처리가 비활성화됩니다.
EnableTimeOut	DINT	GSV SSV	없음	이벤트 테스크의 타임아웃 기능을 활성화 또는 비활성화합니다. • 타임아웃 기능을 비활성화하려면 이 속성을 0 으로 설정합니다. • 타임아웃 기능을 활성화하려면 이 속성을 1(또는 0 이 아닌 모든 값)로 설정합니다.
InhibitTask	DINT	GSV SSV	없음	태스크가 실행되지 않도록 방지함. 태스크가 금지된 경우 컨트롤러가 프로그램 모드에서 실행 또는 테스트 모드로 전환되면 컨트롤러는 계속해서 해당 태스크를 사전 스캔합니다. • 태스크를 활성화하려면 이 속성을 0 으로 설정합니다. • 태스크를 금지(비활성화)하려면 이 속성을 1(또는 0 이 아닌 모든 값)로 설정합니다.

Instance	DINT	GSV	GSV	이 TASK 객체의 인스턴트 수를 제공합니다. 유효한 값은 0 ~ 31 입니다.
LastScanTime	DINT	GSV SSV	없음	프로그램을 마지막 실행했을 때 걸린 시간. 시간은 마이크로초 단위입니다.
MaximumInterval	DINT[2]	GSV SSV	없음	테스크를 연속적으로 실행하는 사이의 최대 시간 범위. DINT[0]에 하위 32 비트 값; DINT[1]에 상위 32 비트 값이 들어 있습니다. 0 값은 테스크를 1 이하로 실행함을 나타냅니다 .
MaximumScanTime	DINT	GSV SSV	없음	이 프로그램에 대해 최대로 기록된 실행 시간. 시간은 마이크로초 단위입니다.
MinimumInterval	DINT[2]	GSV SSV	없음	테스크를 연속적으로 실행하는 사이의 최소 시간 범위. DINT[0]에 하위 32 비트 값; DINT[1]에 상위 32 비트 값이 들어 있습니다. 0 값은 테스크를 1 이하로 실행함을 나타냅니다.
Name	문자열	GSV	GSV	테스크 이름.
OverlapCount	DINT	GSV SSV	GSV SSV	태스크가 실행 중인 상태에서 트리거되는 횟수. 이벤트 또는 주기 태스크에 유효합니다. 카운트를 지우려면 이 속성을 0 으로 설정합니다.
Priority	INT	GSV SSV	GSV	다른 테스크와 비교한 이 테스크의 상대적 우선 순위. 유효값은 0 ~ 15 입니다.
Rate	DINT	GSV SSV	GSV	테스크 실행 시간 범위. 시간은 마이크로초 단위입니다.
StartTime	DINT[2]	GSV SSV	없음	태스크 마지막 실행이 시작된 경우 WALLCLOCKTIME 값. DINT[0]에 하위 32 비트 값; DINT[1]에 상위 32 비트 값이 들어 있습니다.

Status	DINT	GSV SSV	없음	태스크에 대한 상태 정보 제공함. 컨트롤러가 다음 비트 중 하나를 설정하면 수동으로 해당 비트를 지워야 합니다. 평가 내용: • EVENT 명령어가 태스크(이벤트 태스크만)를 트리거한지 여부를 결정하려면 비트 0 을 검사합니다. • 타임아웃이 태스크(이벤트 태스크만) 트리거한지 여부를 결정하려면 비트 1 을 검사합니다. • 태스크에 대해 오버랩이 발생한지 결정하려면 비트 2 를 검사합니다.
Watchdog	DINT	GSV SSV	GSV	이 테스크와 관련된 모든 프로그램 실행을 위한 시간 제한. 시간은 마이크로초 단위입니다. 0 을 입력할 경우 해당 값이 할당됩니다. **시간:** **테스크 유형:** 0.5 초 주기적 5.0 초 연속적

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

TimeSynchronize 객체 액세스

TIMESYNCHRONIZE 객체는 네트워크 측정 및 제어 시스템에 대해 정밀한 클럭 동기화 프로토콜을 위한 IEEE 1588 (IEC 61588) 표준에 대한 CIP 를 제공합니다. GSV/SSV 명령어를 통해 TIMESYNCHRONIZE 객체에 액세스합니다.

이 객체에 대한 자세한 내용은 Integrated Architecture® and CIP Sync Configuration Application Techniques(발행 번호 IA-AT003)을 참조합니다.

속성	데이터 유형	명령어	설명	
ClockType	INT	GSV	클럭 유형.	
			비트	**클럭 유형**
			0	일반 클럭
			1	경계 클럭
			2	피투피 투명 클럭
			3	단대단 투명 클럭
			4	관리 노드
			다른 모든 비트가 예약되어 있습니다.	
CurrentTimeMicroseconds	LINT	GSV	현재 시스템 시간 값(마이크로초)	
CurrentTimeNanoseconds	LINT	GSV	현재 시스템 시간 값(나노초)	
DomainNumber	SINT	GSV	PTP 클럭 도메인. 값은 0 ~ 255 입니다. 기본값은 0 입니다.	
CurrentTimeMicroseconds	LINT	GSV	현재 시스템 시간 값(마이크로초)	
CurrentTimeNanoseconds	LINT	GSV	현재 시스템 시간 값(나노초)	
DomainNumber	SINT	GSV	PTP 클럭 도메인. 값은 0 ~ 255 입니다. 기본값은 0 입니다.	
GrandMasterClockInfo	구조	GSV	그랜드마스터 클럭에 대한 속성 정보. 24 바이트의 저장소가 필요합니다.	
그랜드마스터 클럭 정보 구조:				
ClockIdentity	SINT[8]			
ClockClass	INT			
TimeAccuracy	INT			
OffsetScaledLogVariance	INT			
CurrentUtcOffset	INT			
TimePropertyFlags	INT			
TimeSource	INT			
Priority1	INT			
Priority2	INT			
IsSynchronized	DINT	GSV	로컬 클럭이 마스터에서 동기화됩니다.	
			값	**의미**
			0	동기화되지 않음
			1	동기화됨
LocalClockInfo	구조	GSV	로컬 클럭에 대한 속성 정보. 20 바이트의 저장소가 필요합니다.	

로컬 클럭 정보 구조:			
ClockIdentity	SINT[8]		
ClockClass	INT		
TimeAccuracy	INT		
OffsetScaledLogVariance	INT		
CurrentUtcOffset	INT		
TimePropertyFlags	INT		
TimeSource	INT		
ManufactureIdentity	DINT	GSV	제조업체에 대한 IEEE OUI(Organization Unique Identity).
MaxOffsetFromMaster	LINT	GSV/SSV	마스터의 최대 오프셋(나노초).
MeanPathDelayToMaster	LINT	GSV	마스터 클럭에서 로컬 클럭까지 평균 경로 지연(나노초).
NumberOfPorts	INT	GSV	이 클럭의 포트 수.
OffsetFromMaster	LINT	GSV	최근 동기화 메세지를 기초로 로컬 클럭과 마스터 클럭 사이에 계산된 차이(나노초).
PTPEnable	DINT	GSV/SSV	장치의 CIP Sync/PTP/시간 동기화에 대한 활성화 상태.
			값 / **의미**: 0 / 비활성활; 1 / 활성화
ParentClockInfo	구조	GSV	상위 클럭에 대한 속성 정보. 16 바이트의 저장소가 필요합니다.
상위 클럭 정보 구조:			
ClockIdentity	SINT[8]		
PortNumber	INT		
ObservedOffsetScaledLogVariance	INT		
ObservedPhaseChangeRate	DINT		
PortEnableInfo	구조	GSV	포트가 장치의 각 포트의 구성을 활성화합니다. 크기 = 2 + (활성화 포트 개수 x 4) 최대 크기 = 42 바이트

포트 활성화 상태 구조:			
NumberOfPorts	INT		최대의 포트 수는 10 입니다.
포트 수에 대해 반복된 구조:			
PortNumber	INT		
PortEnable	INT		
PortLogAnnounceIntervalInfo	구조	GSV	장치의 각 PTP 포트에 있는 마스터 클럭에서 발행된 연속적인 “알림”(Announce) 메세지 사이의 범위. 크기 = 2 + (활성화 포트 개수 x 4) 최대 크기 = 42 바이트
포트 로그 알림 범위 구조:			
NumberOfPorts	INT		최대의 포트 수는 10 입니다.
포트 수에 대해 반복된 구조:			
PortNumber	INT		
PortLogAnnounceInterval	INT		
PortLogSyncIntervalInfo	구조	GSV	장치의 각 PTP 포트에 있는 마스터 클럭에서 발행된 연속적인 동기화 메세지 사이의 범위. 크기 = 2 + (활성화 포트 개수 x 4) 최대 크기 = 42 바이트
포트 로그 동기화 범위 구조:			
NumberOfPorts	INT		최대의 포트 수는 10 입니다.
포트 수에 대해 반복된 구조:			
PortNumber	INT		
PortLogAnnounceInterval	INT		
PortPhysicalAddressInfo	구조	GSV	장치에 있는 각 포트의 실제 및 프로토콜 주소. 크기 = 2 + (활성화 포트 개수 x 36) 최대 크기 = 362 바이트

포트의 실제 주소 구조:			
NumberOfPorts	INT		최대의 포트 수는 10 입니다.
포트 수에 대해 반복된 구조:			
PortNumber	INT		
Protocol	SINT[16]		
SizeOfAddress	INT		
Port Address	SINT[16]		
PortProfileIdentityInfo	구조	GSV	장치에 있는 각 포트의 프로파일. 크기 = 2 + (활성화 포트 개수 x 10) 최대 크기 = 102
포트 프로파일 ID 구조:			
NumberOfPorts	INT		최대의 포트 수는 10 입니다.
포트 수에 대해 반복된 구조:			
PortNumber	INT		
ClockIdentity	SINT[8]		
PortProtocolAddressInfo	구조	GSV	장치에 있는 각 포트의 네트워크 및 프로토콜 주소. 크기 = 2 + (활성화 포트 개수 x 22) 최대 크기 = 222
포트 프로토콜 주소 구조:			
NumberOfPorts	INT		최대의 포트 수는 10 입니다.
포트 수에 대해 반복된 구조:			
PortNumber	INT		
NetworkProtocol	INT		
SizeOfAddress	INT		
PortAddress	SINT[16]		
PortStateInfo	구조	GSV	장치에 있는 각 PTP 포트의 현재 상태. 크기 = 2 + (활성화 포트 개수 x 4) 최대 크기 = 42 바이트
포트 상태 구조:			
NumberOfPorts	INT		최대의 포트 수는 10 입니다.
포트 수에 대해 반복된 구조:			
PortNumber	INT		
PortState	INT		
Priority1	SINT	GSV/SSV	로컬 클럭의 Priority1(마스터 오버라이드) 값. **팁**: 서명되지 않은 값입니다.

Priority2	SINT	GSV/SSV	로컬 클럭의 Priority2(타이 브레이커) 값. **팁**: 서명되지 않은 값입니다.
ProductDescription	구조	GSV	클럭이 포함된 장치의 제품 설명. 68 바이트의 저장소가 필요합니다.
제품 설명 구조:			
Size	DINT		
Description	SINT[64]		
RevisionData	구조	GSV	클럭이 포함된 장치의 수정 데이터. 36 바이트의 저장소가 필요합니다.
수정 데이터 구조:			
Size	DINT		
Revision	SINT[32]		
StepsRemoved	INT	GSV	로컬 클럭과 그랜드마스터 클럭 사이의 CIP Sync 지역 개수(경계 클럭 개수 + 1).
SystemTimeAndOffset	구조	GSV	시스템 시간(마이크로초)과 로컬 클럭 값에 대한 오프셋.
시스템 시간 및 오프셋 구조:			
SystemTime	LINT		
SystemOffset	LINT		
UserDescription	구조	GSV	클럭이 있는 장치에 대한 사용자 설명. 132 바이트의 저장소가 필요합니다.
사용자 설명 구조:			
Size	DINT		
Description	SINT[128]		

WallClockTime 객체 액세스

WallClockTime 객체는 컨트롤러가 스케줄링에서 사용하는 타임스탬프를 제공합니다.

팁: WALLCLOCKTIME 객체 설정은 15 초마다 단 1 회 업데이트로 제한됩니다.

중요: GSV 명령어를 사용하여 올바른 시간을 읽으려면 하나의 사용자 태스크에만 WALLCLOCKTIME GSV 를 포함하십시오.

중요: GSV 명령어를 사용하여 올바른 시간을 읽으려면 WALLCLOCKTIME GSV 인스턴스를 기준으로 한 UID/UIE 명령어 쌍을 다른 태스크의 WALLCLOCKTIME GSV 인스턴스에 의해 중단될 수 있는 사용자 태스크에 포함하십시오. WALLCLOCKTIME GSV 가 하나의 사용자 태스크에만 있는 경우에는 UID/UIE 쌍이 필요하지 않습니다.

속성	데이터 유형	명령어	설명
ApplyDST	SINT	GSV SSV	일광 절약 시간제를 활성화할지를 나타냅니다. 값별 의미: • 0. 일광 절약 시간제 조정 불가. • 0 이외의 값. 일광 절약 시간제 조정.
CSTOffset	DINT[2]	GSV SSV	CST(조정 시스템 시간) 객체의 CurrentValue 에서 양의 오프셋. DINT[0]에 하위 32 비트 값; DINT[1]에 상위 32 비트 값이 들어 있습니다. 값의 단위는 미크론입니다. 기본값은 0 입니다.
CurrentValue	DINT[2]	GSV SSV	벽시계 시간 현재값. DINT[0]에 하위 32 비트 값; DINT[1]에 상위 32 비트 값이 들어 있습니다. 1970 년 1 월 1 일 0000 시 이후 경과한 마이크로초 단위의 시간값입니다. CST 및 WALLCLOCKTIME 객체는 컨트롤러 내에서 수학적으로 관련되어 있습니다. 예를 들어, CST CurrentValue 및 WALLCLOCKTIME CTSOffset 을 더한 결과가 WALLCLOCKTIME CurrentValue 입니다.
DateTime	DINT[7]	GSV SSV	날짜 및 시간. 값별 의미: • DINT[0]. 년 • DINT[1]. 월(1 ~ 12) • DINT[2]. 일(1 ~ 31) • DINT[3]. 시간(0 ~ 23) • DINT[4]. 분(0 ~ 59) • DINT[5]. 초(0 ~ 59) • DINT[6]. 마이크로초(0 ~ 999,999)
DSTAdjustment	INT	GSV SSV	일광 절약 시간제를 조정하기 위한 분 수.

LocalDateTime	DINT[7]	GSV SSV	조정된 로컬 현재 시간. 값별 의미: • DINT[0]. 년 • DINT[1]. 월(1 ~ 12) • DINT[2]. 일(1 ~ 31) • DINT[3]. 시간(0 ~ 23) • DINT[4]. 분(0 ~ 59) • DINT[5]. 초(0 ~ 59) • DINT[6]. 마이크로초(0 ~ 999,999)
TimeZoneString	INT	GSV SSV	시간 값에 대한 표준 시간대.

추가 참조

메이저 폴트 유형 및 코드 페이지의 178

마이너 폴트 유형 및 코드 페이지의 186

GSV/SSV 안전 객체

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

안전 태스크의 경우 GSV 및 SSV 명령어가 더욱 제한됩니다.

팁: 안전 및 표준 태스크의 SSV 명령어는 안전 I/O 모듈의 모드 속성에서 비트 0(에러 시 메이저 폴트)을 설정할 수 없습니다.

안전 객체의 경우 다음 표에는 안전 및 표준 태스크에서 GSV 명령어를 사용하여 값을 가져올 수 있는 속성과 SSV 명령어를 사용하여 설정할 수 있는 속성이 나와 있습니다.

주의: GSV/SSV 명령어를 주의 깊게 사용하십시오. 객체를 변경하면 예기치 않은 컨트롤러 작동이나 직원 부상이 발생할 수 있습니다.

안전 객체	속성 이름	속성 설명	안전 태스크에서 액세스 가능		표준 태스크에서 액세스 가능	
			GSV	SSV	GSV	SSV
안전 태스크	Instance	태스크 객체의 인스턴스 번호를 제공함. 유효한 값은 0 ~ 31 입니다.	✔		✔	
	MaximumInterval	태스크의 연속 실행 사이의 최대 시간 간격.			✔	✔
	MaximumScanTime	태스크에 대해 기록된 최대 실행 시간(ms).			✔	✔
	MinimumInterval	태스크의 연속 실행 사이의 최소 시간 간격.			✔	✔
	Priority	다른 태스크와 비교하여 이 태스크의 상대적인 우선 순위. 유효한 값은 0 ~ 15 입니다.	✔		✔	
	Rate	태스크에 대한 기간(ms) 또는 태스크에 대한 타임아웃 값(ms).	✔		✔	
	Watchdog	태스크와 관련된 모든 프로그램 실행에 대한 시간 제한(ms).	✔		✔	
	DisableUpdateOutputs	태스크 종료 시 출력 처리 활성화 또는 비활성화함. • 이 속성을 0 으로 설정하면 태스크 종료 시 출력 처리가 활성화됩니다. • 이 속성을 1(또는 0 이 아닌 모든 값)로 설정하면 태스크 종료 시 출력 처리가 비활성화됩니다.			✔	

안전 객체	속성 이름	속성 설명	안전 태스크에서 액세스 가능		표준 태스크에서 액세스 가능	
	EnableTimeOut	태스크의 타임아웃 기능 활성화 또는 비활성화함. • 타임아웃 기능을 비활성화하려면 이 속성을 0 으로 설정합니다. • 타임아웃 기능을 활성화하려면 이 속성을 1(또는 0 이 아닌 모든 값)로 설정합니다.			✔	
	InhibitTask	태스크가 실행되지 않도록 방지함. 태스크가 금지된 경우 컨트롤러가 프로그램 모드에서 실행 또는 테스트 모드로 전환되면 컨트롤러는 계속해서 해당 태스크를 사전 스캔합니다. • 태스크를 활성화하려면 이 속성을 0 으로 설정합니다. • 태스크를 금지(비활성화)하려면 이 속성을 1(또는 0 이 아닌 모든 값)로 설정합니다.			✔	
	LastScanTime	프로그램을 마지막 실행했을 때 걸린 시간. 시간은 마이크로초 단위입니다.			✔	
	이름(Name)	태스크 이름.				
	OverlapCount	태스크가 실행 중인 상태에서 트리거되는 횟수. 이벤트 또는 주기 태스크에 유효합니다. 카운트를 지우려면 이 속성을 0 으로 설정합니다.			✔	

안전 객체	속성 이름	속성 설명	안전 태스크에서 액세스 가능		표준 태스크에서 액세스 가능	
	StartTime	태스크 마지막 실행이 시작된 경우 WALLCLOCKTIME 값. DINT[0]에 하위 32 비트 값; DINT[1]에 상위 32 비트 값이 들어 있습니다.			✔	
	상태(Status)	태스크에 대한 상태 정보 제공함. 컨트롤러가 다음 비트 중 하나를 설정하면 수동으로 해당 비트를 지워야 합니다. 평가 내용: • EVENT 명령어가 태스크(이벤트 태스크만)를 트리거한지 여부를 결정하려면 비트 0 을 검사합니다. • 타임아웃이 태스크(이벤트 태스크만) 트리거한지 여부를 결정하려면 비트 1 을 검사합니다. • 태스크에 대해 오버랩이 발생한지 결정하려면 비트 2 를 검사합니다.			✔	
안전 프로그램	Instance	프로그램 객체의 인스턴스 번호 제공함.	✔		✔	
	MajorFaultRecord	프로그램에 대한 메이저 폴트 기록함.	✔	✔	✔	
	MaximumScanTime	프로그램에 대해 기록된 최대 실행 시간(ms).			✔	✔
	Disable Flag	프로그램의 실행을 제어함. 값별 의미: • 0. 실행이 활성화됨. • 0 이외의 값. 실행이 비활성화됨.			✔	

안전 객체	속성 이름	속성 설명	안전 태스크에서 액세스 가능		표준 태스크에서 액세스 가능	
	MaximumScanTime	프로그램에 대해 기록된 최대 실행 시간(ms).			✔	
	Minor Fault Record	프로그램에 대한 마이너 폴트를 기록함.			✔	
	LastScanTime	프로그램을 마지막 실행했을 때 걸린 시간. 시간은 마이크로초 단위입니다.			✔	
	이름(Name)	태스크 이름.				
Safety Routine	Instance	루틴 객체의 인스턴스 번호 제공함. 유효한 값은 0 ~ 65,535 입니다.	✔			
Safety Controllers	SafetyLockedState(SINT)	컨트롤러의 안전 잠금이 잠겼는지 또는 잠금 해제되었는지 나타냄.			✔	
	SafetySILConfiguration(SINT)	안전 SIL 구성을 다음과 같이 지정함. • 2 = SIL2/PLd • 3 = SIL3/PLe			✔	

안전 객체	속성 이름	속성 설명	안전 태스크에서 액세스 가능		표준 태스크에서 액세스 가능	
	SafetyStatus(INT) (Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용 가능).	SIL3/PLe 에 대해 구성된 응용 프로그램의 경우 안전 상태를 다음과 같이 지정하십시오. • 안전 태스크 정상 (1100000000000000) • 안전 태스크 작동 불가 (1100000000000011) • 파트너 누락 (0100000000000000) • 파트너 사용할 수 없음 (0100000000000001) • 하드웨어 호환되지 않음(0100000000000010) • 펌웨어 호환되지 않음 (0100000000000011) **팁:** SIL2/PLd 에 대해 구성된 응용 프로그램의 경우 비트 15, 0 및 1 이 1 차 Controller 의 슬롯 + 1 을 기준으로 다른 값일 수 있는 경우 이러한 비트는 무시해야 합니다. 의미는 위의 상태를 참조하십시오. SIL2/PLd 에 대해 구성된 응용 프로그램의 경우 안전 태스크를 다음과 같이 지정하십시오. • 안전 태스크 정상 (x1000000000000xx) • 안전 태스크 작동 불가 (x10000000000001xx)			✔	

안전 객체	속성 이름	속성 설명	안전 태스크에서 액세스 가능		표준 태스크에서 액세스 가능	
	SafetyStatus(INT) (Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러에만 적용 가능).	안전 상태를 다음과 같이 지정함. • 안전 태스크 정상 (100000000000000) • 안전 태스크 작동 불가 (1000000000000001) • 파트너 누락 (0000000000000000) • 파트너 사용할 수 없음 (0000000000000001) • 하드웨어 호환되지 않음(0000000000000010) • 펌웨어 호환되지 않음 (0000000000000011)			✔	
	SafetySignatureExists(SINT)	안전 서명이 존재함을 나타냄.	✔		✔	
	SafetySignatureID(DINT) (Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러에만 적용 가능)	32 비트 ID 번호.			✔	
	SafetySignature(문자열) (Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러에만 적용 가능)	ID 번호 + 날짜 및 타임스탬프.			✔	
	SafetyTaskFaultRecord(DINT)	안전 태스크 폴트를 기록합니다.			✔	
	SafetySignatureIDLong SINT [33] (Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용 가능)	첫 번째 바이트는 안전 서명 ID 의 크기(바이트 단위로)이고 나머지 32 바이트에는 32 바이트 안전 서명 ID 의 내용이 포함되어 있습니다.			✔	

안전 객체	속성 이름	속성 설명	안전 태스크에서 액세스 가능		표준 태스크에서 액세스 가능	
	SafetySignatureIDHex(문자열) (Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용 가능)	서명 ID 의 64 자 16 진수 문자열 표현.			✔	
	SafetySignatureDateTime (문자열) (Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용 가능)	안전 서명의 27 자 날짜 시간(mm/dd/yyyy, hh:mm:ss.iii<AM 또는 PM> 형식)			✔	

추가 참조

입력/출력 명령어 페이지의 165

상태 플래그 모니터링

컨트롤러는 특정 이벤트 모니터링 로직에서 사용할 수 있는 상태 키워드를 지원합니다.

- 상태 키워드의 경우 *대소문자를 구분하지 않습니다.*
- 상태 플래그는 빠르게 변화하므로 Logix Designer 응용 프로그램에는 플래그 상태가 *표시되지 않습니다*(즉, 상태 플래그가 설정된 경우에도 그 플래그를 참조하는 명령어가 강조 표시되지 않습니다).
- 키워드에 태그 앨리어스를 *정의할 수 없습니다.*

다음 키워드를 사용할 수 있습니다.

평가 내용:	사용:
다음 중 하나에 해당하기 때문에 저장할 값이 대상에 맞지 않는지 • 대상 최대값보다 큼 또는 • 대상 최소값보다 작음 **중요:** S:V 가 해제에서 설정으로 전환될 때마다 마이너 폴트가 생성됩니다(유형 4, 코드 4)	S:V
명령어의 대상 값이 0 인지	S:Z

명령어의 대상 값이 음인지	S:N
산술 연산으로 인해 데이터 유형 이외의 비트를 사용하려고 하는 자리올림 또는 차용이 발생하는지 예: • 3 + 9 합으로 1 의 자리올림이 발생함 • 25 - 18 빼기로 10 의 차용이 발생함	S:C
현재 프로그램의 루틴을 이용한 첫 번째 정상 스캔인지	S:FS
한 개 이상의 마이너 폴트가 발생했는지 • 프로그램 실행으로 인해 마이너 폴트가 발생하는 경우 컨트롤러에서 이 비트를 설정합니다. • 프로그램 실행가 관련 없는 마이너 폴트 발생 시(예: 저전력) 컨트롤러에서 이 비트를 설정하지 않습니다.	S:MINOR

메세지 유형 선택

MSG 명령어를 입력하고 MESSAGE 구조를 지정한 후 메세지 구성(Message Configuration) 대화 상자의 구성(Configuration) 탭을 클릭하여 메세지 세부 정보를 지정합니다.

구성(Configuration) 탭에는 .TO 비트의 설정/해제 확인란도 있습니다.

구성하는 세부 정보는 선택한 메세지 유형에 따라 다릅니다.

대상 장치가 다음에 해당하는 경우:	**다음 메세지 유형 중 하나를 선택:**
LOGIX 5000 컨트롤러	CIP 데이터 표 읽기(CIP data table read) CIP 데이터 표 쓰기(CIP data table write)
Logix Designer 응용 프로그램을 사용하여 구성한 I/O 모듈	모듈 재구성(Module Reconfigure) CIP 일반(CIP Generic)
PLC-5® 컨트롤러	PLC-5 유형 읽기(PLC-5 typed read) PLC-5 유형 쓰기(PLC-5 typed write) PLC-5 워드 범위 읽기(PLC-5 word range read) PLC-5 워드 범위 쓰기(PLC-5 word range write)

SLC™ 컨트롤러 MicroLogix™ 컨트롤러	SLC 유형 읽기(SLC typed read) SLC 유형 쓰기(SLC typed write)
블록 전송 모듈	블록 전송 읽기(block transfer read) 블록 전송 쓰기(block transfer write)
PLC-3® 프로세서	PLC-3 유형 읽기(PLC-3 typed read) PLC-3 유형 쓰기(PLC-3 typed write) PLC-3 워드 범위 읽기(PLC-3 word range read) PLC-3 워드 범위 쓰기(PLC-3 word range write)
PLC-2® 프로세서	PLC-2 보호되지 않은 읽기(PLC-2 unprotected read) PLC-2 보호되지 않은 쓰기(PLC-2 unprotected write)

다음 구성 정보를 지정해야 합니다.

해당 필드:	**지정 내용:**
소스 요소(Source Element)	읽기 메세지 유형을 선택하는 경우 소스 요소는 대상 장치에서 읽을 데이터의 주소입니다. 대상 장치의 주소 지정 구문을 사용하십시오. 쓰기 메세지 유형을 선택하는 경우 Source 태그는 대상 장치로 전송할 첫 번째 태그 요소입니다. I/O 구조 태그와 부울 값은 사용할 수 없지만, 그 외 모든 데이터 유형(예: INT, DINT)은 사용할 수 있습니다.
요소 수(Number of Elements)	읽거나 쓰는 요소의 수는 메세지 유형 및 사용하려는 데이터 유형에 따라 다릅니다. "워드 범위" 및 "보호되지 않은" 메세지의 경우 요소 크기가 대화 상자에 표시됩니다. CIP 및 "유형" 메세지의 경우 요소는 쓰기 소스 또는 읽기 대상으로 지정하는 배열의 단일 요소입니다.
대상 요소(Destination Element)	읽기 메세지 유형을 선택하는 경우 Destination 태그는 대상 장치에서 읽은 데이터를 저장할 LOGIX 5000 컨트롤러의 첫 번째 태그 요소입니다. 쓰기 메세지 유형을 선택하는 경우 대상 요소는 대상 장치에서 데이터를 쓸 위치의 주소입니다.

추가 참조

CIP 메세지 지정하기 페이지의 318

PLC-5 메세지 지정하기 페이지의 326

SLC 메세지 지정하기 페이지의 208

블록 전송 메세지 지정하기 페이지의 208

PLC-3 메세지 지정하기 페이지의 325

PLC-2 메세지 지정하기 페이지의 327

모듈 폴트: 16#0000 - 16#00ff

다음은 모듈 폴트입니다. 16#0000 - 16#00ff

코드	문자열	설명 및 예상 원인/해결 방법
16#0001	연결 오류입니다.	Module 에 대한 연결이 실패했습니다.
16#0002	리소스를 사용할 수 없습니다.	다음 이유로 인해 발생합니다. • 연결하는 데 사용할 Controller 또는 통신 Module 에 대해 사용 가능한 연결이 충분하지 않습니다. Controller 또는 통신 Module 에 사용하는 연결을 확인하십시오. 연결을 모두 사용한 경우 사용한 연결 일부를 해제하거나 다른 Module 을 추가하여 잘못된 연결을 라우팅합니다. • Controller 의 I/O 메모리 제한을 초과했습니다. 사용 가능한 I/O 메모리를 확인하여 필요한 경우 프로그램이나 Tag 를 변경합니다. • 대상 I/O Module 에 사용 가능한 연결이 충분하지 않습니다. 이 I/O Module 에 대한 연결을 만드는 Controller 의 수를 확인하고 연결 수가 I/O Module 제한 내에 있는지 확인합니다.
16#0005	연결 요청 오류: 잘못된 클래스입니다.	Controller 가 Module 과의 연결을 시도하는 동안 오류가 발생했습니다. 다음 이유로 인해 발생합니다. • Module 에 대한 연결의 구성 주소가 잘못되었습니다. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다. 1756-DHRIO Module 을 사용하는 경우 소프트웨어에서 선택된 채널 유형(DH+ 또는 원격 I/O 네트워크)이 Module 의 회전 스위치 설정과 일치하는지 확인하십시오.

16#0006	연결 요청 오류: 잘못된 클래스입니다.	다음 이유로 인해 발생합니다. • 응답 버퍼가 너무 작아 응답 데이터를 처리하지 못합니다. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다.
16#0007	연결 요청 오류: 잘못된 클래스입니다.	서비스 요청에 연결되지 않았지만 연결되어야 합니다.
16#0008	서비스 요청 오류: 지원되지 않는 서비스입니다.	Controller 가 Module 로부터 Module 에서 지원하지 않는 서비스 요청을 시도하는 중입니다.
16#0009	Module 구성이 잘못됨: 파라메터 오류입니다. **팁:** 이 폴트에 대한 추가 폴트 정보가 연결 탭에 16 진수 코드로 표시됩니다.	Module 에 대한 구성이 잘못되었습니다. Module 구성이 데이터 모니터에서 또는 프로그램 방식으로 변경되었을 수 있습니다. Module 에 대해 사용 가능한 경우 추가 오류 코드를 위해 Module 속성 대화 상자의 연결 탭에 액세스합니다. 추가 오류 코드는 오류를 발생시키는 구성 파라메터를 나타냅니다. 이 오류가 지워지고 연결이 제대로 설정되려면 여러 파라메터를 수정해야 할 수 있습니다.
16#000A	Get_Attributes_List 또는 Set_Attributes_List 의 속성이 0 이 아닌 상태임	다음 이유로 인해 발생합니다. • 연결 유형이 잘못된 연결을 만드는 중입니다. • 개체 속성 또는 Tag 값이 잘못되었습니다. 개체 속성 또는 Tag 가 잘못된 경우 Logix Designer 파일을 내보낸 다음 다시 가져옵니다. 해당하는 경우 다시 가져온 후 ControlNet 네트워크를 다시 예약합니다.
16#000C	서비스 요청 오류: 서비스 요청에 대한 모드/상태가 잘못되었습니다.	Controller 가 Module 로부터 서비스를 요청하는 동안 오류가 발생했습니다. 우선 Module 에 오류가 없는지 확인하십시오. I/O Module 의 경우 이 오류는 Module 의 상태가 다음 중 하나에 해당할 수 있습니다. • 제한된 통신이지만 Major Fault 가 있습니다. • 펌웨어 업데이트를 완료해야 하거나 현재 완료 중입니다. 정확한 원인에 대한 내용은 Module 정보 탭을 참조하십시오.

16#000D	개체가 이미 존재합니다.	해당 인스턴스가 이미 사용 중인 I/O 맵 인스턴스가 만들어집니다.
16#000E	속성 값을 설정할 수 없습니다.	변경할 수 없는 속성 값을 변경하도록 MSG 명령어가 구성되었습니다.
16#000F	요청된 서비스에 대해 액세스 권한이 거부되었습니다.	삭제할 수 없는 맵 개체를 삭제하도록 MSG 명령어가 구성되었습니다.
16#0010	Module 의 모드 또는 상태에서 개체가 요청된 서비스를 수행할 수 없습니다.	장치의 상태가 서비스 요청 처리를 차단합니다.
16#0011	응답 데이터가 너무 큽니다.	메시지에 대한 응답의 데이터 크기가 대상에 비해 너무 큽니다. 대상을 반환되는 데이터 크기 및 유형을 처리할 수 있는 Tag 로 변경합니다.
16#0013	Module 구성이 거부됨: 데이터 크기가 너무 작습니다.	Module 의 구성이 잘못되었습니다 - 부족한 구성 데이터가 전송되었습니다. 대상이 올바른 Module 인지 확인하십시오.
16#0014	정의되지 않거나 지원되지 않는 속성입니다.	존재하지 않는 속성을 변경하도록 MSG 명령어가 구성되었습니다.
16#0015	Module 구성이 거부됨: 데이터 크기가 너무 큽니다.	Module 의 구성이 잘못되었습니다 - 너무 많은 구성 데이터가 전송되었습니다. 대상이 올바른 Module 인지 확인하십시오.

모듈 폴트: 16#0100 - 16#01ff

다음은 모듈 폴트입니다. 16#0100 - 16#01ff

코드	문자열	설명 및 예상 원인/해결 방법
16#0100	연결 요청 오류: Module 이 사용 중입니다.	• 액세스 중인 연결이 이미 사용 중입니다. 다음 이유로 인해 발생합니다. • Controller 가 Module 에 대해 특정 연결을 시도 중이며, Module 에서는 이 연결을 둘 이상 지원할 수 없습니다. • 연결의 대상에서 소유자가 이미 실행 중인 연결을 다시 만들려고 하는지 인식합니다.

16#0103	서비스 요청 오류: CIP 전송 클래스가 지원되지 않습니다.	다음 이유로 인해 발생합니다. • Controller 가 Module 에서 지원되지 않는 서비스를 요청하고 있습니다. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다.
16#0106	연결 요청 오류: 다른 Controller 에서 Module 을 소유하고 구성했습니다. Unicast 가 사용되는 경우 Module 이 하나의 연결만 수락할 수 있습니다.	연결에 대한 소유권 충돌이 발생했습니다. 다음 조건 중 하나가 존재합니다. • 다른 소유자(예: 다른 Controller)와의 소유권 충돌로 인해 이 Module 과의 연결 요청이 거부되었습니다. 이 오류는 그 출력을 구성하고 제어하는 데 단일 소유자만을 허용하는 출력 Module 과 같은 Module 에서 발생할 수 있습니다. 이 오류는 또한 Module 이 수신 전용으로 구성되어 있고 하나의 연결만 지원하는 경우에도 발생할 수 있습니다. • 소유자가 이더넷/IP 를 통해 유니캐스트 연결을 사용하여 Module 에 연결되어 있는 경우 소유자가 하나의 연결만 제어하므로 Module 에 대한 다른 연결이 실패합니다. 소유자가 이더넷/IP 를 통해 멀티캐스트 연결을 사용하여 Module 에 연결되어 있는 경우 소유자가 하나의 연결만 제어하므로 Module 에 대한 유니캐스트 연결이 실패합니다. 소유자 및 수신 전용 연결을 모두 멀티캐스트로 구성합니다.
16#0107	연결 요청 오류: 알 수 없는 유형입니다.	액세스 중인 연결을 찾을 수 없습니다.

16#0108	연결 요청 오류: 연결 유형(멀티캐스트/유니캐스트)이 지원되지 않습니다.	Controller 가 Module 에서 지원되지 않는 연결 유형을 요청하고 있습니다. 다음 조건 중 하나가 존재합니다. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. • 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 키 지정 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다. • Unicast 연결을 사용하도록 Consumed Tag 를 구성했을 수 있지만 제작 Controller 가 Unicast 연결을 지원하지 않습니다.
16#0109	연결 요청 오류: 잘못된 연결 크기입니다. **팁:** 이 오류에 대한 추가 오류 정보가 오류가 있는 연결 인스턴스 번호와 연관된 Tag 이름으로 표시됩니다.	연결 크기가 예상과 일치하지 않습니다. 다음 이유로 인해 발생합니다. • Controller 가 Module 과의 연결을 설정하는 중이지만 연결할 수 없습니다. 연결 크기가 잘못되었습니다. • Controller 가 이 Controller 의 Tag 와 크기가 일치하지 않는 제작 Controller 의 Tag 에 연결을 시도하는 중일 수 있습니다. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. • 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 키 지정 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다. Module 이 1756 ControlNet Module 인 경우 섀시 크기가 맞는지 확인하십시오. 원격 I/O 어댑터의 경우 랙 크기 및/또는 랙 밀도가 올바른지 확인하십시오.

16#0110	연결 요청 오류: Module 이 구성되지 않았습니다.	Controller 가 Module 과의 수신 전용 연결을 설정하는 중이지만 연결할 수 없습니다. 소유자(예: 다른 Controller)에 의해 Module 이 구성 및 연결되지 않았습니다. 이 Controller 는 Module 구성을 필요로 하지 않는 수신 전용 연결을 시도하는 중이므로 이 Module 의 소유자가 아닙니다. 이 Module 에 소유자가 구성되고 연결되기 전에는 연결할 수 없습니다.
16#0111	RPI(Requested Packet Interval)가 범위를 벗어났습니다.	다음 이유로 인해 발생합니다. • 지정된 RPI(Requested Packet Interval)가 이 Module 또는 이 Module 경로의 Module 에 대해 잘못되었습니다. 제작자의 RPI 를 사용하려면 고급 탭을 참조하십시오. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다. • 수신 전용 연결의 경우: 이 Module 의 소유자에 의해 설정된 RPI 가 요청된 RPI 보다 느립니다. 요청된 RPI 를 높이거나 소유자 Controller 가 사용 중인 RPI 를 낮추십시오. 유효한 RPI 값은 Module 속성 대화 상자의 연결 탭을 참조하십시오.
16#0113	연결 요청 오류: Module 연결 제한을 초과했습니다.	연결 수가 Module 에서 사용할 수 있는 개수보다 큽니다. 연결 수를 줄이거나 하드웨어를 업그레이드해야 합니다. 연결 수를 줄이려면: • Flex I/O 통신 어댑터 통신 형식을 입력 또는 출력 구성에서 **랙 최적화**로 변경합니다. 통신 형식이 변경되면 I/O 구성 트리에서 어댑터를 제거하고 다시 작성해야 합니다. • 구성에서 ControlNet 을 통한 메시징을 사용하는 경우 동시에 실행 중인 개수를 줄이도록 메시지 순서를 지정하거나 메시지 개수를 줄입니다. 메시지(MSG 명령어)도 연결을 사용합니다.

16#0114	전자 키 지정 불일치: 전자 키 지정 제품 코드 및/또는 공급업체 ID 가 일치하지 않습니다.	실제 Module 하드웨어의 제품 코드가 소프트웨어에서 생성된 Module 의 제품 코드와 일치하지 않습니다. 이 Module 에 대한 전자 키 지정이 실패했습니다. 소프트웨어에서 생성된 Module 과 실제 Module 하드웨어 사이에 불일치가 있을 수 있습니다.
16#0115	전자 키 지정 불일치: 전자 키 지정 제품 유형이 일치하지 않습니다.	실제 Module 하드웨어의 제품 유형이 소프트웨어에서 생성된 Module 의 제품 유형과 일치하지 않습니다. 이 Module 에 대한 전자 키 지정이 실패했습니다. 소프트웨어에서 생성된 Module 과 실제 Module 하드웨어 사이에 불일치가 있을 수 있습니다.
16#0116	전자 키 지정 불일치: 주 수정 버전 및/또는 부 수정 버전이 잘못되었거나 올바르지 않습니다.	Module 의 주 수정 버전 및/또는 부 수정 버전이 소프트웨어에서 생성된 Module 의 주 수정 버전 및/또는 부 수정 버전과 일치하지 않습니다. 호환 Module 또는 정확히 일치 키 지정을 선택한 경우 올바른 주 수정 버전과 부 수정 버전을 지정했는지 확인하십시오. 이 Module 에 대한 전자 키 지정이 실패했습니다. 소프트웨어에서 생성된 Module 과 실제 Module 하드웨어 사이에 불일치가 있을 수 있습니다.
16#0117	연결 요청 오류: 잘못된 연결 지점입니다. **팁:** 이 오류에 대한 추가 오류 정보가 오류가 있는 C2C(Controller to Controller)와 연관된 Tag 이름으로 표시됩니다.	잘못된 포트 또는 이미 사용 중인 포트에 연결되었습니다. 다음 조건 중 하나가 존재합니다. • 다른 Controller 가 이 Module 을 소유하고 이 Controller 가 선택하지 않은 다른 통신 형식: I/O Module 로 연결되었습니다. 선택한 통신 형식이 Module 의 첫 번째 소유자 Controller 가 선택한 통신 형식과 동일한지 확인하십시오. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다. • Controller 는 제작 Controller 의 존재하지 않는 Tag 에 대해 연결을 시도하는 중일 수 있습니다.

16#0118	Module 구성이 거부됨: 형식 오류입니다.	잘못된 구성 형식이 사용되었습니다. 다음 조건 중 하나가 존재합니다. • 지정한 구성 클래스가 Module 에서 지정한 클래스와 일치하지 않습니다. • 연결 인스턴스를 Module 에서 인식할 수 없습니다. • 연결에 지정한 경로가 일치하지 않습니다. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다.
16#0119	연결 요청 오류: Module 소유자가 없습니다.	제어 연결이 열리지 않습니다. 수신 전용 연결이 요청되면 제어 연결이 열리지 않습니다.
16#011A	연결 요청 오류: 연결 리소스가 부족합니다.	Controller 가 Module 과의 연결을 설정하는 중이지만 연결할 수 없습니다 - 필요한 리소스를 사용할 수 없습니다. Module 이 1756 ControlNet Module 인 경우, 최대 5 개의 Controller 가 Module 에 대해 랙 최적화 연결을 구성할 수 있습니다. 이 숫자를 초과하지 않았는지 확인하십시오. Module 이 1794-ACN15, 1794-ACNR15 또는 1797-ACNR15 어댑터인 경우 하나의 Controller 만 Module 에 대해 랙 최적화 연결을 구성할 수 있습니다. 이 숫자를 초과하지 않았는지 확인하십시오.

모듈 폴트: 16#0200 - 16#02ff 다음은 모듈 폴트입니다. 16#0200 - 16#02ff.

코드	문자열	설명 및 예상 원인/해결 방법
16#0203	연결 시간이 초과되었습니다.	소유자 또는 주관자는 대상 장치가 네트워크 또는 백플레인에 있지만 I/O 데이터 및 메시지가 응답하지 않음을 인식합니다. 즉, 대상에 연결될 수는 있지만 그 응답이 예상과 다릅니다. 예를 들어 이 오류는 멀티캐스트 이더넷 패킷을 반환하지 않는 위치를 나타낼 수 있습니다. 이 오류가 발생하면 Controller 는 일반적으로 연결 제거와 다시 만들기를 연속으로 시도합니다. FLEX I/O Module 을 사용 중인 경우 올바른 터미널 장치를 사용 중인지 확인합니다.
16#0204	연결 요청 오류: 연결 요청 시간이 초과되었습니다.	Controller 가 연결을 시도 중이지만 대상 Module 이 응답하지 않습니다. 또한 이 장치가 백플레인이나 네트워크에서 누락된 것으로 나타납니다. 복구하려면 다음 조치를 취합니다. • Module 이 제거되지 않고 아직 작동하고 있으며 전원을 공급받고 있는지 확인하십시오. • 올바른 슬롯 번호를 지정했는지 확인하십시오. • Module 이 제대로 네트워크에 연결되었는지 확인하십시오. FLEX I/O Module 을 사용 중인 경우 올바른 터미널 블록을 사용하는지 확인하십시오.
16#0205	연결 요청 오류: 잘못된 파라메터입니다.	다음 이유로 인해 발생합니다. • Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다 - 파라메터에 오류가 있습니다. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다.

16#0206	연결 요청 오류: 요청된 크기가 너무 큽니다.	다음 이유로 인해 발생합니다. • Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다 - 요청 크기가 너무 큽니다. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다.

모듈 폴트: 16#0300 - 16#03ff

다음은 모듈 폴트입니다. 16#0300 - 16#03ff

코드	문자열	설명 및 예상 원인/해결 방법
16#0301	연결 요청 오류: 버퍼 메모리가 부족합니다.	다음 조건 중 하나가 존재할 수 있습니다. • Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다. 경로의 Module 에 메모리가 부족합니다. • Controller 가 제작되도록 표시되지 않은 제작 Controller 의 Tag 에 대해 연결을 시도하는 중일 수 있습니다. • Controller 는 제작 Controller 의 Tag 에 대해 연결을 시도하는 중일 수 있습니다. 이 Tag 는 충분한 컨수머를 허용하도록 구성할 수 없습니다. • 이 Module 을 통한 연결의 크기 또는 수를 줄이십시오. • Module 과 Controller 사이의 네트워크 Module 중 하나에 메모리가 부족할 수 있습니다. 시스템의 네트워크 구성을 확인하십시오. • Module 의 메모리가 부족할 수 있습니다. 시스템 구성과 Module 의 기능을 확인하십시오. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에

		사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다.
16#0302	연결 요청 오류: 통신 대역폭이 부족합니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다 - 경로의 Module 이 통신 대역폭 용량을 초과했습니다. RPI(Requested Packet Interval)를 늘리고 RSNetWorx 를 통해 네트워크를 재구성하십시오. 다른 브리지 Module 에 로드를 분배하십시오.
16#0303	연결 요청 오류: 사용할 수 있는 브리지가 없습니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다 - 경로의 Module 이 통신 대역폭 용량을 초과했습니다. 다른 브리지 Module 에 로드를 분배하십시오.
16#0304	예약된 데이터를 보내도록 구성되지 않았습니다.	ControlNet Module 이 데이터를 전송하도록 예약되어 있지 않습니다. RSNetWorx for ControlNet 소프트웨어를 사용하여 ControlNet 네트워크를 예약하거나 다시 예약합니다.
16#0305	연결 요청 오류: Controller 의 ControlNet 구성이 브리지의 구성과 일치하지 않습니다.	Controller 의 ControlNet 구성이 브리지 Module 의 구성과 일치하지 않습니다. 이는 네트워크가 예약된 후 ControlNet Module 이 변경되었기 때문이거나 새 Control Program 이 Controller 에 로드되었기 때문입니다. RSNetWorx for ControlNet 소프트웨어를 사용하여 연결을 다시 예약합니다.
16#0306	CCM(ControlNet Configuration Master)을 사용할 수 없습니다.	CCM(ControlNet Configuration Master)을 찾을 수 없습니다. 1756-CNB 와 PLC-5C Module 만 CCM 이 될 수 있는 Module 이며, CCM 은 1 번 노드가 되어야 합니다. 1756-CNB 또는 PLC-5C Module 이 1 번 노드에 있으며 올바르게 작동하는지 확인하십시오. 시스템의 전원을 켤 때 이 오류가 일시적으로 발생할 수 있으며 CCM 이 위치하게 되면 해결됩니다.
16#0311	연결 요청 오류: 잘못된 포트입니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다. I/O 구성 트리의 모든 Module 이 올바른 Module 인지 확인합니다.
16#0312	연결 요청 오류: 잘못된 링크 주소입니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다 - 잘못된 링크 주소가 지정되었습니다. 슬롯 번호, 네트워크 주소 또는 원격 I/O 섀시 번호 및 시작 그룹을 링크

		주소로 사용할 수 있습니다. 이 Module 에 대해 선택한 슬롯 번호는 랙의 크기보다 크지 않아야 합니다. ControlNet 노드 번호는 RSNetWorx for ControlNet 소프트웨어에서 네트워크에 대해 구성된 최대 노드 번호보다 크지 않아야 합니다.
16#0315	연결 요청 오류: 잘못된 세그먼트 유형입니다.	세그먼트 유형 또는 경로가 잘못되었습니다. 다음 이유로 인해 발생합니다. • Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다. 연결 요청이 잘못되었습니다. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다.
16#0317	연결 요청 오류: 연결이 예약되지 않았습니다.	Controller 가 ControlNet 연결을 설정하는 동안 Module 에서 오류가 발생했습니다. RSNetWorx for ControlNet 소프트웨어를 사용하여 이 Module 에 대한 연결을 예약하거나 다시 예약합니다.
16#0318	연결 요청 오류: 잘못된 링크 주소 - 자신에게 라우팅할 수 없습니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다. 링크 주소가 잘못되었습니다. 연결된 ControlNet Module 에 올바른 슬롯 번호 및/또는 노드 번호가 선택되었는지 확인하십시오.
16#0319	연결 요청 오류: 중복된 섀시에서 사용할 수 있는 보조 리소스가 없습니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다 - 중복 Module 에 연결을 지원하기 위해 필요한 리소스가 없습니다. 이 Module 을 통한 연결의 크기 또는 수를 줄이거나 시스템에 다른 Controller 또는 ControlNet Module 을 추가하십시오.

16#031a	연결 요청 오류: 랙 연결이 거부되었습니다.	Controller 가 Module 과의 직접 연결을 설정하는 동안 오류가 발생했습니다. 동일한 섀시의 1756-CNB/R 를 통해 랙 최적화된 연결이 이미 이 Module 에 대해 구성되어 있습니다. • 동일한 섀시의 1756-CNB/R 을 통해 이 Module 에 연결합니다. • 직접 연결을 사용하기 위해 다른 1756-CNB/R 을 통해 이 Module 에 연결합니다. • 랙 최적화에서 직접으로의 첫 번째 연결을 변경한 후 두 번째 직접 연결을 다시 설정합니다. • Module 의 역할을 하는 동일한 섀시의 Controller 로부터 이 Module 에 연결합니다(1756-CNB/R 을 통해 연결하지 않음).
16#031e	연결 요청 오류: Tag 를 사용할 수 없습니다.	• Controller 가 제작 Controller 의 Tag 에 연결을 시도하는 동안 오류가 발생했습니다. • Controller 가 제작 Controller 의 Tag 에 연결을 시도하는 중이며 이 Tag 는 이미 너무 많은 컨수머가 사용하고 있습니다. Tag 에 대한 최대 컨수머 수를 늘리십시오.
16#031f	연결 요청 오류: Tag 를 사용할 수 없습니다.	기호 인스턴스에 해당하는 SC(Servicing Controller) 연결 개체를 찾을 수 없습니다.
16#0322	연결 요청 오류: 연결 지점 불일치	연결 지점 불일치가 발생했습니다. 다음 이유로 인해 발생합니다. • 요청한 새 연결이 기존 연결과 일치하지 않습니다. 연결을 사용 중인 Controller 를 검사하여 모든 구성이 동일한지 확인합니다. • 요청한 연결이 수신자 또는 제어 연결 유형이 아닙니다.

모듈 폴트: 16#0800 - 16#08ff

다음은 모듈 폴트입니다.

코드	문자열	설명 및 예상 원인/해결 방법
16#0800	Module 경로의 네트워크 링크가 오프라인 상태입니다.	사용할 수 있는 해석이 없습니다.
16#0801	호환되지 않는 멀티캐스트 RPI 입니다.	사용할 수 있는 해석이 없습니다.

16#0810	대상 응용 프로그램 데이터를 사용할 수 없습니다.	제어 응용 프로그램이 대상 장치에 의해 제작될 데이터를 초기화하지 않았습니다. 이는 "데이터 전송" 연결이 대상 장치에 구성되고 대상 장치용 제어 응용 프로그램이 제작될 데이터를 초기화하지 않았을 때 발생할 수 있습니다. 이 연결 오류를 보고한 "데이터 전송" 연결과 관련된 대상 장치에 대해 제어 응용 프로그램을 시작하고 데이터의 쓰기를 1 회 이상 수행합니다. 이 동작의 수행 방법에 대한 내용은 대상 장치와 해당 제어 응용 프로그램에 대한 설명서를 참조하십시오.
16#0814	연결 요청 오류: Data Type 이 일치하지 않습니다.	잘못된 연결 상태 정보가 발견되었습니다.

모듈 폴트: 16#fd00 - 16#fdff

모듈 폴트: 16#fd00 - 16#fdff

코드	문자열	설명 및 예상 원인/해결 방법
16#fd03	연결 요청 오류: 필요한 연결이 없습니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다 - 이 Module 은 연결의 특정 집합과 연결 유형을 필요로 하며, 이 연결 유형 중 하나가 없습니다. • 기술 지원으로 연락하십시오. • http:/www.support.rockwellautomation.com
16#fd04	연결 요청 오류: CST 마스터가 검색되지 않았습니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다 - 이 Module 은 섀시의 CST 마스터를 필요로 합니다. • 이 섀시의 Module(일반적으로 Controller)을 CST 마스터로 구성하십시오. • 기술 지원으로 연락하십시오. • http:/www.support.rockwellautomation.com
16#fd05	연결 요청 오류: Axis 또는 그룹이 할당되지 않았습니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다 - 이 Module 은 할당된 Axis 또는 그룹 테이블을 필요로 합니다. • 그룹 또는 Axis 를 할당합니다. • 기술 지원으로 연락하십시오. • http:/www.support.rockwellautomation.com
16#fd06	전환 오류	SERCOS 링을 새 Phase 로 전환하기 위한 Controller 명령으로 인해 Module 에서 오류가 반환되었습니다. 드라이브 노드가 중복되었는지 확인합니다.

16#fd07	잘못된 SERCOS 데이터 속도입니다.	SERCOS 링을 구성하지 못했습니다. 모든 장비에 대한 Baud Rate 가 동일해야 하며 드라이브 및 SERCOS Module 에서 지원되어야 합니다.
16#fd08	SERCOS 통신 오류	통신 오류는 주로 2 가지 오류, 즉 물리적 오류와 인터페이스 오류로 인해 발생할 수 있습니다. 물리적 오류의 예상 원인은 다음과 같습니다. • 링이 끊어짐 • 커넥터가 느슨함 • 광섬유가 깨끗하지 않음 • 부적합한 드라이브 접지로 인한 전기 노이즈 • 링에 노드가 너무 많음 인터페이스 오류는 타사 드라이브를 구성할 때 발생합니다. 인터페이스 오류의 예상 원인은 다음과 같습니다. • SERCOS MST 가 없음(프로토콜 오류) • AT 가 누락됨(예상한 시기에 드라이브에서 데이터를 보내지 않았음) • Phase 3 에서의 SERCOS 타이밍 오류 • SERCOS Module 로 반환된 드라이브 데이터의 오류
16#fd09	노드 초기화 오류	Controller 에서 주기적 작업을 위한 노드를 구성하려는 시도로 인해 오류가 반환되었습니다.
16#fd0a	Axis 속성 오류	모션 Module 에서 잘못된 응답이 수신되었습니다.
16#fd0c	오류: 다른 그랜드마스터 오류	최종 장치에 Controller 와 다른 그랜드마스터가 있습니다.
16#fd1f	잘못된 안전 프로토콜 형식	안전 네트워크 세그먼트를 경로에 추가하는 중 오류가 발생했습니다.
16#fd20	안전 Task 없음	실행 중인 안전 Task 가 나타나지 않습니다.
16#fd22	섀시 크기 불일치	Controller 에 대해 구성된 물리적 확장 I/O Module 의 수를 확인한 다음 Controller 속성 대화 상자의 일반 페이지에 있는 확장 I/O 목록에서 선택한 Module 의 수를 업데이트합니다.
16#fd23	섀시 크기 초과	Controller 가 지원하는 물리적 확장 I/O 의 수를 확인하려면 Controller 속성 대화 상자를 열고 일반 페이지에서 확장 I/O 목록을 확장합니다. 물리적 확장 I/O Module 의 수를 확장 I/O 목록의 선택과 일치하도록 구성합니다.

모듈 폴트: 16#fe00 - 16#feff

모듈 폴트: 16#fe00 - 16#feff.

코드	문자열	설명 및 예상 원인/해결 방법
16#fe01		잘못된 구성 형식이 발생했습니다.
16#fe02	RPI 가 범위를 벗어났습니다.	지정된 RPI 가 이 모듈에 대해 잘못되었습니다. • 유효 RPI 값은 연결(Connection) 탭을 참조하십시오.
16#fe03		입력 연결 지점이 설정되지 않았습니다.
16#fe04	연결 요청 에러: 잘못된 입력 데이터 포인터입니다.	컨트롤러가 모듈과의 연결을 설정하는 동안 에러가 발생했습니다.
16#fe05	연결 요청 에러: 잘못된 입력 데이터 크기입니다.	다음 중 하나: • 컨트롤러가 모듈과의 연결을 설정하는 동안 에러가 발생했습니다. • 사용 중인 모듈(즉, 물리적 모듈)이 I/O 구성 트리에 지정된 모듈과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 폴트는 모듈이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 비활성화 또는 호환 모듈 옵션이 모듈 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 모듈에 I/O 구성 트리에서 지정된 모듈과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 모듈을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 모듈과 정확히 일치하는지 확인합니다.
16#fe06		입력 강제 적용 지점이 설정되지 않았습니다.
16#fe07		출력 연결 지점이 설정되지 않았습니다.
16#fe08	연결 요청 에러: 잘못된 출력 데이터 포인터입니다.	컨트롤러가 모듈과의 연결을 설정하는 동안 에러가 발생했습니다.

16#fe09	연결 요청 에러: 잘못된 출력 데이터 크기입니다.	다음 중 하나: • 컨트롤러가 모듈과의 연결을 설정하는 동안 에러가 발생했습니다. • 사용 중인 모듈(즉, 물리적 모듈)이 I/O 구성 트리에 지정된 모듈과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 폴트는 모듈이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 비활성화 또는 호환 모듈 옵션이 모듈 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 모듈에 I/O 구성 트리에서 지정된 모듈과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 모듈을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 모듈과 정확히 일치하는지 확인합니다.
16#fe0a		출력 강제 적용 포인터가 설정되지 않았습니다.
16#fe0b	유효하지 않은 기호 문자열.	다음 중 하나: • 이 모듈에서 사용할 태그가 잘못되었습니다. 태그가 생산 중으로 표시되었는지 확인하십시오. • 사용 중인 모듈(즉, 물리적 모듈)이 I/O 구성 트리에 지정된 모듈과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 폴트는 모듈이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 비활성화 또는 호환 모듈 옵션이 모듈 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 모듈에 I/O 구성 트리에서 지정된 모듈과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 모듈을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 모듈과 정확히 일치하는지 확인합니다.
16#fe0c	유효하지 않은 PLC-5 인스턴스 번호입니다.	컨트롤러가 PLC-5 와의 연결을 설정하는 동안 에러가 발생했습니다. 지정한 인스턴스 번호가 PLC-5 에 지정된 속성인지 확인합니다.
16#fe0d	피어 컨트롤러에 태그가 없습니다.	기호 인스턴스 번호가 설정되지 않았습니다.
16#fe0e	자동 펌웨어 업데이트 진행 중입니다.	현재 모듈을 업데이트하는 중입니다.

16#fe0f	자동 펌웨어 업데이트 실패: 펌웨어 파일이 모듈과 호환되지 않습니다.	펌웨어 슈퍼바이저가 지원되지 않는 모듈을 업데이트하려고 했습니다.
16#fe10	자동 펌웨어 업데이트 실패: 펌웨어 파일을 찾을 수 없습니다.	모듈을 업데이트하기 위한 펌웨어 파일을 찾을 수 없습니다.
16#fe11	자동 펌웨어 업데이트 실패: 펌웨어 파일이 잘못되었습니다.	펌웨어 파일이 손상되었습니다.
16#fe12	자동 펌웨어 업데이트가 실패했습니다.	모듈을 업데이트하는 동안 에러가 발생했습니다.
16#fe13	자동 펌웨어 업데이트 실패: 활성 연결이 감지되었습니다.	대상 모듈에 활성 연결을 구성할 수 없습니다.
16#fe14	자동 펌웨어 업데이트 보류 중: NVS 파일에서 해당 모듈 ID 를 검색 중입니다.	현재 펌웨어 파일을 읽는 중입니다.
16#fe22		대상-발신자 네트 파라미터 연결 유형이 유효하지 않습니다.
16#fe23		대상-발신자 네트 파라미터 연결이 유니캐스트 허용 여부를 지정하지 않습니다.

모듈 폴트: 16#ff00 - 16#ffff

다음은 모듈 폴트입니다. 16#ff00 - 16#ffff.

코드	문자열	설명 및 예상 원인/해결 방법
16#ff00	연결 요청 오류: 연결 인스턴스가 없습니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다. 물리적 Module 이 소프트웨어에서 생성된 것과 같은 Module 유형(또는 호환 Module)인지 확인하십시오. Module 이 ControlNet 네트워크를 통해 연결된 원격 섀시의 1756-DHRIO Module 인 경우 네트워크가 RSNetWorx 소프트웨어를 통해 예약되었는지 확인하십시오. RSNetWorx for ControlNet 소프트웨어를 통해 네트워크가 예약된 후에도 온라인 상태이며 1756-DHRIO Module 이 DH+ 네트워크만을 위해 구성된 경우 #ff00 Module 오류(연결 인스턴스 없음)가 발생할 수 있습니다. Module 속성 대화 상자에 Module 의 상태로 오류 발생이 표시되지만 Module 이 올바르게 통신하고 있습니다. 오류 메시지와 오류 상태를 무시하고 계속하십시오.
16#ff01	연결 요청 오류: Module 경로가 너무 깁니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다. 이 Module 경로의 길이가 올바른지 확인하십시오.

16#ff04		잘못된 상태에서 원격 Controller 의 맵 인스턴스가 연결에 액세스하려고 했습니다.
16#ff08	연결 요청 오류: Module 에 대한 경로가 잘못되었습니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다. 이 Module 경로의 길이가 올바른지 확인하십시오.
16#ff0b	Module 구성이 잘못됨: 잘못된 형식입니다.	다음 이유로 인해 발생합니다. • Module 에 대한 구성이 잘못되었습니다. • 사용 중인 Module(즉, 물리적 Module)이 I/O 구성 트리에 지정된 Module 과 다르기 때문에 연결 또는 서비스가 실패합니다. 이 오류는 Module 이 전자 키 지정 테스트를 통과한 경우에도 발생할 수 있습니다. 이러한 현상은 정확히 일치 옵션 대신 키 지정 사용 안 함 또는 호환 Module 옵션이 Module 구성에 사용된 경우 발생합니다. 전자 키 지정 테스트를 통과했지만 연결 중인 Module 에 I/O 구성 트리에서 지정된 Module 과 다른 기능이나 설정이 있어 시도 중인 연결 또는 서비스를 지원하지 않습니다. 사용 중인 Module 을 검사하여 Logix Designer 응용 프로그램의 I/O 구성 트리에 지정된 Module 과 정확히 일치하는지 확인합니다.
16#ff0e	연결 요청 오류: 브리지에서 수락된 연결이 없습니다.	Controller 가 Module 과의 연결을 설정하는 동안 오류가 발생했습니다.

CIP 메세지 지정하기

CIP 데이터 표 읽기 및 쓰기 메세지 유형은 LOGIX 5000 컨트롤러 간에 데이터를 전송합니다.

다음 명령 선택	실행할 작업
CIP 데이터 표 읽기(CIP Data Table Read)	다른 컨트롤러에서 데이터 읽기. Source 및 Destination 유형이 일치해야 합니다.
CIP 데이터 표 쓰기(CIP Data Table Write)	다른 컨트롤러에 데이터를 쓰기. Source 및 Destination 유형이 일치해야 합니다.

I/O 모듈 재구성

모듈 재구성 메세지를 사용하여 새 구성 정보를 I/O 모듈로 보냅니다.

재구성 중 다음 작업이 발생합니다.

- 입력 모듈이 계속해서 컨트롤러로 입력 데이터를 보냅니다.
- 출력 모듈이 계속해서 출력 장치를 제어합니다.

모듈 재구성 메세지에는 다음 구성 속성이 필요합니다.

다음 속성에서	선택할 항목
메세지 유형(Message Type)	모듈 재구성(Module Reconfigure)

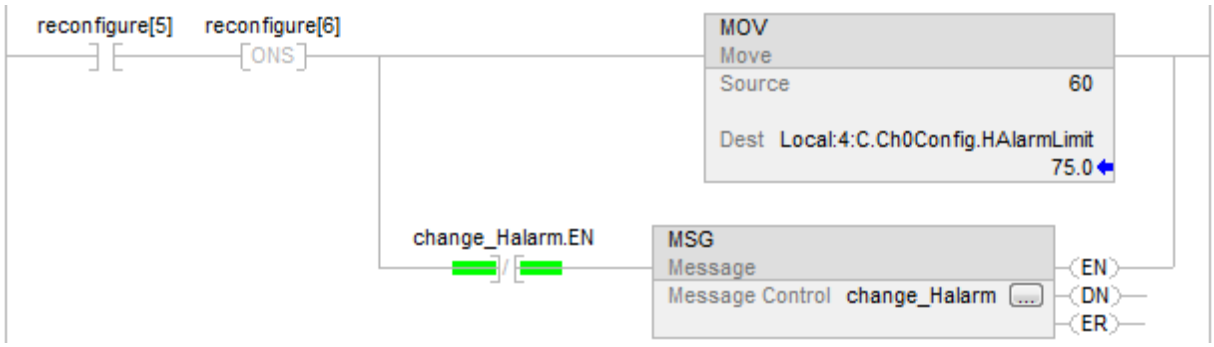
예

I/O 모듈을 재구성하려면 아래 단계를 따르십시오.

1. 모듈의 구성 태그에 필요한 구성원을 새 값으로 설정합니다.
2. 모듈 재구성 메세지를 모듈로 보냅니다.
   reconfigure[5]가 설정되면 슬롯 4 의 로컬 모듈에 대해 상한 알람을 60 으로 설정합니다. 그런 다음 모듈 재구성 메세지가 모듈로 새 알람 값을 전송합니다. 원샷 명령어는 reconfigure[5]가 설정되어 있는 경우 렁이 여러 메세지를 모듈에 보내지 않도록 합니다.

   팁: MSG.EN 비트의 XIO 를 연속 MSG 렁 사전 조건으로 항상 포함하는 것이 좋습니다.

릴레이 래더



ST(스트럭처드 텍스트)

```
IF reconfigure[5] AND NOT reconfigure[6]THEN
Local:4:C.Ch0Config.HAlarmLimit := 60;

IF NOT change_Halarm.EN THEN MSG(change_Halarm);

END_IF; END_IF;

reconfigure[6] := reconfigure[5];
```

CIP 일반 메세지 지정하기

중요: ControlLogix 모듈에는 MSG 명령어를 사용하고 CIP 일반 메세지 유형을 선택하여 호출할 수 있는 서비스가 있습니다.

실행할 작업	다음 속성에서	입력 또는 선택할 항목	
디지털 출력 모듈에서 펄스 테스트 수행	메세지 유형(Message Type)	CIP 일반(CIP Generic)	
	서비스 유형(Service Type)	펄스 테스트(Pulse Test)	
	소스(Source)	유형 INT [5]의 tag_name	
		이 어레이에 포함된 항목	설명
		tag_name[0]	테스트할 포인트의 비트 마스크(한 번에 한 포인트만 테스트)
		tag_name[1]	예약됨, 0 으로 둠
		tag_name[2]	펄스 너비(수백 µ, 일반적으로 20)
		tag_name[3]	ControlLogix I/O 에 대한 0 점 교차 지연(수백 µ, 일반적으로 40)
		tag_name[4]	지연 확인
	대상(Destination)	비워두기	
감사 값 가져오기	메세지 유형(Message Type)	CIP 일반(CIP Generic)	
	서비스 유형(Service Type)	감사 값 가져오기	
	소스 요소(Source Element)	이 필드는 변경할 수 없음, 비어 있음	
	소스 길이(Source Length)	이 필드는 변경할 수 없음, 0 바이트로 설정	
	대상 요소(Destination Element)	이 어레이에 포함된 항목	설명

		유형 DINT[2]의 Tag_name 또는 LINT	이 태그에는 컨트롤러에 대한 감사 값이 포함되어 있습니다. **중요:** Allen-Bradley® 컨트롤러에서 LINT 데이터 유형을 사용하는 경우 Rockwell Automation 에서는 DINT[2] 데이터 유형을 사용하여 제한 사항을 피할 것을 권장합니다.
변경 사항이 있는지 모니터링된 컨트롤러 이벤트 가져오기	메세지 유형(Message Type)	CIP 일반(CIP Generic)	
	서비스 유형(Service Type)	감지할 변경 사항 가져오기(Changes to Detect Get)	
	소스 요소(Source Element)	이 필드는 변경할 수 없음, 비어 있음	
	소스 길이(Source Length)	이 필드는 변경할 수 없음, 0 바이트로 설정	
	대상 요소(Destination Element)	이 어레이에 포함된 항목	설명
		유형 DINT[2]의 Tag_name 또는 LINT	이 태그는 컨트롤러에 대해 모니터링되는 변경 사항의 비트 마스크를 나타냅니다. **중요:** Allen-Bradley 컨트롤러에서 LINT 데이터 유형을 사용하는 경우 Rockwell Automation 에서는 DINT[2] 데이터 유형을 사용하여 제한 사항을 피할 것을 권장합니다.
변경 사항이 있는지 모니터링되는 컨트롤러 이벤트 설정	메세지 유형(Message Type)	CIP 일반(CIP Generic)	
	서비스 유형(Service Type)	감지할 변경 사항 설정(Changes to Detect Set)	
	소스 요소(Source Element)	이 어레이에 포함된 항목	설명
		유형 DINT[2]의 Tag_name 또는 LINT	이 태그는 컨트롤러에 대해 모니터링되는 변경 사항의 비트 마스크를 나타냅니다. **중요:** Allen-Bradley 컨트롤러에서 LINT 데이터 유형을 사용하는 경우 Rockwell Automation 에서는 DINT[2] 데이터 유형을 사용하여 제한 사항을 피할 것을 권장합니다.

	소스 길이(Source Length)	이 필드는 변경할 수 없음, 8 바이트로 설정
	대상 요소(Destination Element)	이 필드는 변경할 수 없음, 비어 있음
디지털 출력 모듈에서 전기 퓨즈 리셋	메세지 유형(Message Type)	CIP 일반(CIP Generic)
	서비스 유형(Service Type)	전기 퓨즈 리셋(Reset Electronic Fuse)
	소스(Source)	DINT 유형의 태그 이름 이 태그는 퓨즈를 리셋할 지점의 비트 마스크를 나타냅니다.
	대상(Destination)	비워두기
디지털 입력 모듈에서 래치된 진단 리셋	메세지 유형(Message Type)	CIP 일반(CIP Generic)
	서비스 유형(Service Type)	래치된 진단 리셋(I)(Reset Latched Diagnostics (I))
	소스(Source)	DINT 유형의 tag_name 이 태그는 진단을 리셋할 지점의 비트 마스크를 나타냅니다.
디지털 출력 모듈에서 래치된 진단 리셋	메세지 유형(Message Type)	CIP 일반(CIP Generic)
	서비스 유형(Service Type)	래치된 진단 리셋(O)(Reset Latched Diagnostics (O))
	소스(Source)	DINT 유형의 tag_name 이 태그는 진단을 리셋할 지점의 비트 마스크를 나타냅니다.
아날로그 입력 모듈의 알람 래치 해제	메세지 유형(Message Type)	CIP 일반(CIP Generic)
	서비스 유형(Service Type)	래치 해제할 알람을 선택하십시오. • 모든 알람 래치 해제(I)(Unlatch All Alarms (I)) • 아날로그 상한 알람 래치 해제(I)(Unlatch Analog High Alarm (I)) • 아날로그 상-상한 알람 래치 해제(I)(Unlatch Analog High High Alarm (I)) • 아날로그 하한 알람 래치 해제(I)(Unlatch Analog Low Alarm (I)) • 아날로그 하-하한 알람 래치 해제(I)(Unlatch Analog Low Low Alarm (I)) • 속도 알람 래치 해제(I)(Unlatch Rate Alarm (I))
	인스턴스(Instance)	래치 해제할 알람 채널
아날로그 출력 모듈의 알람	메세지 유형(Message Type)	CIP 일반(CIP Generic)

래치 해제	서비스 유형(Service Type)	래치 해제할 알람을 선택하십시오. • 모든 알람 래치 해제(O)(Unlatch All Alarms (O)) • 상한 알람 래치 해제(O)(Unlatch High Alarm (O)) • 하한 알람 래치 해제(O)(Unlatch Low Alarm (O)) • 램프 알람 래치 해제(O)(Unlatch Ramp Alarm (O))
	인스턴스(Instance)	래치 해제할 알람 채널

변경 비트 정의에 대해 모니터링되는 컨트롤러 이벤트 가져오기/설정

태그 이름	데이터 유형	비트 정의
변경 사항이 있는지 모니터링된 컨트롤러 이벤트 가져오기 변경 사항이 있는지 모니터링되는 컨트롤러 이벤트 설정	DINT[0]	각 비트에는 특정한 의미가 있습니다. 0 Logix Designer 응용 프로그램을 통해 이동식 미디어에 저장 1 온라인 편집이 수락, 테스트 또는 어셈블되었음 2 부분적 가져오기 온라인 전환이 완료되었음 3 SFC 강제 적용이 활성화되었음 4 SFC 강제 적용이 비활성화되었음 5 SFC 강제 적용이 제거되었음 6 SFC 강제 적용이 수정되었음 7 I/O 강제 적용이 활성화되었음 8 I/O 강제 적용이 비활성화되었음 9 I/O 강제 적용이 제거되었음 10 I/O 강제 적용이 변경되었음 11 연결되지 않은 소스에서 펌웨어 업데이트 12 이동식 미디어를 통해 펌웨어 업데이트 13 워크스테이션을 통해 모드 변경 14 모드 스위치를 통해 모드 변경 15 메이저 폴트가 발생했음 16 메이저 폴트가 해제됐음 17 모드 스위치를 통해 메이저 폴트가 해제되었음 18 태스크 속성이 수정되었음 19 프로그램 속성이 수정되었음 20 컨트롤러 타임 슬라이스 옵션이 수정되었음 21 이동식 미디어가 분리되었음 22 이동식 미디어가 삽입되었음

		23 안전 서명이 생성되었음 24 안전 서명이 삭제되었음 25 안전 잠금 26 안전 잠금 해제 27 상수 태그 값이 변경되었음 28 상수 태그 다중 값이 변경되었음 29 상수 태그 속성이 해제되었음 30 태그가 상수로 설정되었음 31 사용자 지정 로그 항목이 추가되었음
	DINT[1]	32 상관 관계에 영향을 미치는 변경 33 실행 모드 서명 보호 속성 설정 지원 34 실행 모드 서명 보호 속성 해제 지원 35 ~ 63 사용되지 않음

팁:

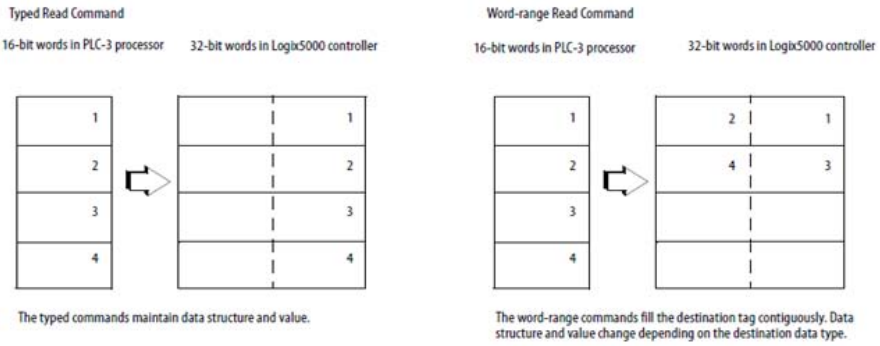
- **CIP 일반**(CIP Generic) 메세지 유형을 선택하면 **통신**(Communication) 탭에서 **대형 연결**(Large Connection) 옵션을 사용할 수 있습니다. 메세지가 480 바이트보다 큰 경우 대형 CIP 일반 연결을 사용합니다. 500 바이트가 일반적이지만 메세지 앞에는 헤더가 있습니다. 대형 CIP 연결은 최대 3980 바이트의 메세지용입니다.
- 이 **대형 연결**(Large Connection) 확인란은 **연결됨**(Connected) 상자를 선택한 경우에만 사용할 수 있으며 **CIP 일반**(CIP Generic)은 **구성**(Configuration) 탭의 메세지 유형으로 선택됩니다.
- **대형 연결**(Large Connection) 옵션은 Logix Designer 응용 프로그램 버전 21.00.00 이상 및 RSLogix 5000 소프트웨어 버전 20.00.00 이상에서만 사용할 수 있습니다.

PLC-3 메세지 지정하기

PLC-3 메세지 유형은 PLC-3 프로세서용으로 설계된 것입니다.

선택해야 하는 명령:	목적:
PLC3 유형화된 읽기(PLC3 Typed Read)	정수 또는 REAL 유형 데이터를 읽습니다. 정수의 경우, 이 명령은 PLC-3 프로세서의 16 비트 정수를 읽어서 LOGIX 5000 컨트롤러의 SINT, INT, 또는 DINT 데이터 배열에 저장함으로써 데이터 무결성을 유지합니다. 이 명령은 또한 PLC-3 의 부동 소수점 데이터를 읽어서 LOGIX 5000 컨트롤러의 REAL 데이터 유형 태그에 저장합니다.
PLC3 유형화된 쓰기(PLC3 Typed Write)	정수 또는 REAL 유형 데이터를 씁니다. 이 명령은 SINT 또는 INT 데이터를 PLC-3 정수 파일에 옮겨 쓰고 데이터 무결성을 유지합니다. DINT 데이터가 INT 데이터 유형(−32,768 ≥ 데이터 ≤ 32,767)에 부합되는 한 이를 쓸 수 있습니다.. 이 명령은 또한 LOGIX 5000 컨트롤러의 REAL 유형 데이터를 PLC-3 부동 소수점 파일로 옮겨 쓸 수 있습니다.
PLC3 워드 범위 읽기(PLC3 Word Range Read)	PLC-3 메모리에서 데이터 유형에 상관없이 일정 범위의 연속된 16 비트 워드를 읽습니다. 이 명령은 소스 요소로 지정된 주소에서 시작하여 요청된 16 비트 워드의 수를 차례로 읽습니다. 소스 요소의 데이터는 대상 태그로 지정된 주소에서 시작하여 저장됩니다.
PLC3 워드 범위 쓰기(PLC3 Word Range Read)	LOGIX 5000 메모리에서 데이터 유형에 상관없이 연속된 범위의 16 비트 워드를 PLC-3 메모리로 씁니다. 이 명령은 소스 태그로 지정된 주소에서 시작하여 요청된 16 비트 워드의 수를 차례로 읽습니다. 소스 태그의 데이터는 PLC-3 프로세서에서 대상 요소(Destination Element)로 지정된 주소에서 시작하여 저장됩니다.

아래 다이어그램에서는 유형화된 명령과 워드 범위 명령이 어떻게 다른지 보여줍니다. 예에서는 PLC-3 프로세서에서 LOGIX 5000 컨트롤러로 읽어 들이는 명령을 사용합니다.



PLC-5 메세지 지정하기

PLC-5 컨트롤러와 통신하려면 PLC-5 메세지 유형을 사용합니다.

선택해야 하는 명령:	목적:
PLC-5 유형화된 읽기(PLC-5 Typed Read)	16 비트 정수, 부동 소수점 또는 문자열 유형 데이터를 읽고 데이터 무결성을 유지합니다.
PLC-5 유형화된 쓰기(PLC-5 Typed Write)	16 비트 정수, 부동 소수점 또는 문자열 유형 데이터를 써서 데이터 무결성을 유지합니다.
PLC-5 워드 범위 읽기(PLC-5 Word Range Read)	PLC-5 메모리에서 데이터 유형에 상관없이 연속된 범위의 16 비트 워드를 읽습니다. 이 명령은 소스 요소로 지정된 주소에서 시작하여 요청된 16 비트 워드의 수를 차례로 읽습니다. 소스 요소의 데이터는 대상 태그로 지정된 주소에서 시작하여 저장됩니다.
PLC-5 워드 범위 쓰기(PLC-5 Word Range Write)	LOGIX 5000 메모리에서 데이터 유형에 상관없이 연속된 범위의 16 비트 워드를 PLC-5 메모리에 씁니다. 이 명령은 소스 태그로 지정된 주소에서 시작하여 요청된 16 비트 워드의 수를 차례로 읽습니다. 소스 태그의 데이터는 PLC-5 프로세서에서 대상 요소로 지정된 주소에서 시작하여 저장됩니다.

PLC-5 유형화된 읽기 및 유형화된 쓰기 메세지에 대한 데이터 유형

다음 표에서는 PLC-5 유형화된 읽기 및 PLC-5 유형화된 쓰기 메세지와 함께 사용할 데이터 유형을 보여 줍니다.

PLC-5 데이터 유형:	사용할 LOGIX 5000 데이터 유형:
B	INT
F	REAL
N	INT
	DINT(값이 ≥ −32,768 이고 ≤ 32,767 인 경우에만 PLC-5 컨트롤러에 DINT 값을 옮겨 씀)
S	INT
ST	STRING

유형화된 읽기와 유형화된 쓰기 명령은 또한 SLC 5/03 프로세서(OS303 이상), SLC 5/04 프로세서(OS402 이상), SLC 5/05 프로세서에서도 실행할 수 있습니다.

PLC-2 메세지 지정하기

PLC-2 메세지 유형은 PLC-2 프로세서용으로 설계된 것입니다.

선택해야 하는 명령:	목적:
PLC2 비보호 읽기(PLC2 Unprotected Read)	PLC-2 데이터 표의 모든 영역 또는 다른 프로세서의 PLC-2 호환성 파일에서 16 비트 워드를 읽어 들입니다.
PLC2 비보호 쓰기(PLC2 Unprotected Write)	PLC-2 데이터 표의 모든 영역 또는 다른 프로세서의 PLC-2 호환성 파일에서 16 비트 워드를 옮겨 씁니다.

메세지 전송은 16 비트 워드를 사용하므로 LOGIX 5000 태그는 전송된 데이터(일반적으로 INT 배열임)를 적절하게 저장해야 합니다.

비교 명령어

비교 명령어

비교 명령어를 통해 식 또는 특정 비교 명령어를 사용하여 값을 비교할 수 있습니다.

사용 가능한 명령어

래더 다이어그램

CMP	EQU	GEQ	GRT	LEQ	LES	LIM	MEQ	NEQ

함수 블록 다이어그램

FBD 블록

EQU	GEQ	GRT	LEQ	LES	LIM	MEQ	NEQ

FBD 평션

=f	≥f	>f	≤f	<f	LIMf	MEQf	≠f
EQU	GEQ	GRT	LEQ	LES	LIM	MEQ	NEQ

ST(스트럭쳐드 텍스트)

사용할 수 없음

실행할 작업:	사용할 명령어:
식을 기반으로 값을 비교합니다.	CMP
두 값이 동일한지 테스트합니다.	EQU
한 값이 두 번째 값보다 크거나 같은지 테스트합니다.	GEQ
한 값이 두 번째 값보다 큰지 테스트합니다.	GRT
한 값이 두 번째 값보다 작거나 같은지 테스트합니다.	LEQ
한 값이 두 번째 값보다 작은지 테스트합니다.	LES

한 값이 다른 두 값 사이에 있는지 테스트합니다.	LIM
마스크를 통해 두 값을 전달하고 동일한지 테스트합니다.	MEQ
한 값이 두 번째 값과 다른지 테스트합니다.	NEQ

부동 소수점과 정수와 같이 데이터 유형이 다른 값을 비교합니다.

볼드체 데이터 유형은 최적의 데이터 유형을 나타냅니다. 명령어의 모든 파라미터가 동일한 최적의 데이터 유형(일반적으로 DINT 또는 REAL)을 사용하는 경우 명령어는 가장 적은 메모리를 사용하면서 가장 빠르게 실행됩니다.

추가 참조

계산/연산 명령어 페이지의 411

비교(CMP)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

연산자, 태그 및 즉시 값을 사용하여 CMP 식을 정의합니다. 괄호()를 사용하여 더 복잡한 식의 섹션을 정의합니다.

CMP 명령어를 사용할 경우 복잡한 식을 하나의 명령어에 담을 수 있다는 장점이 있습니다.

식을 평가할 때 다음 조건 중 하나라도 참인 경우 계산이 수행되기 전 REAL 외 모든 피연산자가 REAL 로 변환됩니다.

- 식의 모든 피연산자가 REAL 입니다.
- 이 식에는 SIN, COS, TAN, ASN, ACS, ATN, LN, LOG, DEG, RAD 가 포함됩니다.

안전 응용 프로그램에서 허용 가능한 연산자에 대한 규칙이 있습니다. *유효 연산자* 섹션을 참조하십시오.

사용 가능한 언어

래더 다이어그램

펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

피연산자

다음은 CMP 명령어에 대한 피연산자입니다.

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

다음은 래더 다이어그램 피연산자입니다.

피연산자	데이터 유형	형식	설명
Expression	SINT INT DINT REAL 문자열 유형	즉시 태그	태그 및/또는 연산자로 구분된 즉시 값으로 구성된 식

식의 형식

식에 사용된 연산자마다 하나 또는 두 개의 피연산자(태그 또는 즉시 값)를 입력해야 합니다. 다음 표를 사용하여 식 내의 연산자 및 피연산자 형식을 지정합니다.

연산자 작용 대상:	사용 형식:	예
피연산자 1 개	연산자(피연산자)	ABS(tag)
피연산자 2 개	operand_a 연산자 operand_b	tag_b + 5 tag_c AND tag_d (tag_e**2) MOD (tag_f / tag_g)

연산 순서 정의

식의 연산은 표시된 순으로 할 필요없이 규정된 순서로 명령어에 의해 수행됩니다. 괄호 내에 조건을 그룹화, 명령어 자체의 연산에 앞서 괄호 안의 연산을 수행하게 하는 등으로 연산 순서를 지정할 수 있습니다.

연산 순서가 같은 경우에는 왼쪽에서 오른쪽 순으로 수행됩니다.

순서	연산
1	( )
2	ABS, ACS, ASN, ATN, COS, DEG, FRD, LN, LOG, RAD, SIN, SQR, TAN, TOD, TRN
3	**
4	- (부정), NOT
5	*, /, MOD
6	- (빼기), +
7	AND
8	XOR
9	OR
10	<, <=, >, >=, =, <>

식에 문자열 사용

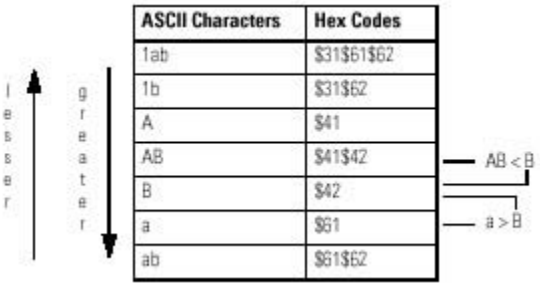
식에 ASCII 문자로 된 문자열을 사용하려면 다음 지침을 따르십시오.

- 식으로 두 문자열 태그를 비교합니다.
- ASCII 문자를 직접 식 안에 입력할 수 없습니다.
- 아래의 연산자를 사용할 수 있습니다.

연산자	설명
=	같음
<	보다 작음
<=	보다 작거나 같음
>	보다 큼
>=	보다 크거나 같음
<>	같지 않음

- 문자들이 일치하면 문자열이 같습니다.

- ASCII 문자는 대소문자가 구분됩니다. 대문자 A($41)와 소문자 a($61)는 다릅니다.
- 16 진수 문자값으로 한 문자열이 다른 문자열보다 작거나 큰지를 판단합니다.
- 두 문자열이 전화번호부와 마찬가지로 정렬된 경우 문자열의 순서로 어느 문자열이 큰지 알 수 있습니다.



연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	아니요
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	식에 연산 상태 플래그에 영향을 미치는 연산자(예: +, −, *, /)가 포함되어 있는 경우 CMP 명령어가 연산 상태 플래그에 영향을 미칩니다.

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

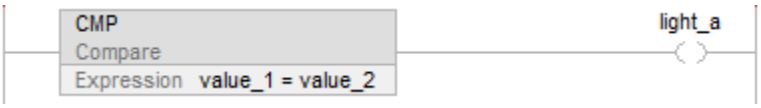
실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A.
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 식이 거짓으로 평가되는 경우 렁-출력-조건이 거짓으로 해제됨
사후 스캔	N/A.

예

래더 다이어그램



value_1 과 value_2 가 같은 경우 light_a 는 참으로 설정됩니다. value_1 과 value_2 가 같지 않으면 light_a 는 거짓으로 해제됩니다.

추가 참조

비교 명령어 페이지의 329

유효한 연산자 페이지의 407

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

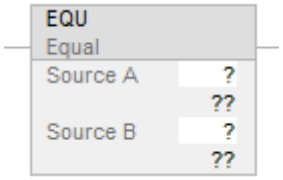
같음(EQU)

이 명령어는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

활성화된 경우, EQU 명령어와 = 연산자는 소스 A 와 소스 B 가 같은지 테스트합니다.

사용 가능한 언어

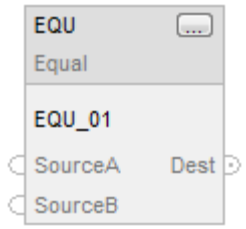
래더 다이어그램



함수 블록 다이어그램

평션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 평션

팁: FBD 평션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 '=' 연산자를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

숫자 비교

피연산자	데이터 유형 **CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러**	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	형식	설명
Source A	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source B 와 비교해 테스트할 값
Source B	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source A 와 비교해 테스트할 값

문자열 비교

팁: 즉시 문자열 리터럴은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

피연산자	데이터 유형	형식	설명
Source A	문자열 유형	즉시 리터럴 값 태그	Source B 와 비교해 테스트할 문자열
Source B	문자열 유형	즉시 리터럴 값 태그	Source A 와 비교해 테스트할 문자열

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
EQU	FBD_COMPARE	태그	EQU 구조

FBD_COMPARE 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	REAL	SourceB 와 비교해 테스트할 값
SourceB	REAL	SourceA 와 비교해 테스트할 값

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었는지를 나타냅니다.
Dest	BOOL	SourceA 가 SourceB 와 동일한 경우 참으로 설정합니다. SourceA 가 SourceB 와 동일하지 않으면 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형	설명
SourceA(상위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceB 와 대조하여 검사할 값.
SourceB(하위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceA 와 비교해 테스트할 값

출력 피연산자(오른쪽 핀)	데이터 유형	설명
Dest	BOOL	SourceA 가 SourceB 와 동일한 경우 참으로 설정합니다. SourceA 가 SourceB 와 동일하지 않으면 거짓으로 해제됩니다.

*FBD 펑션*을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

폴트에 대해서는 *EQU 문자열 비교 흐름도*를 참조하십시오.

배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정

렁-입력-조건이 참임	**숫자 비교:**
	Source A 및 Source B 가 NAN 이 아니고
	Source A 와 Source B 가 같은 경우.
	  렁-출력-조건을 참으로 설정
	else
	  렁-출력-조건을 거짓으로 해제

	문자열 비교:
	EQU 문자열 비교 흐름도를 참조하십시오.
	출력이 거짓인 경우
	  렁-출력-조건을 거짓으로 해제
	else
	  렁-출력-조건을 참으로 설정

사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정

EnableIn 이 참임	**숫자 비교:**
	EnableOut 에서 EnableIn 으로 설정
	SourceA 와 SourceB 가 NAN 이 아니고
	SourceA 와 SourceB 가 같은 경우.
	  Dest 를 참으로 설정
	else
	  Dest 를 거짓으로 해제

명령어 최초 실행	N/A

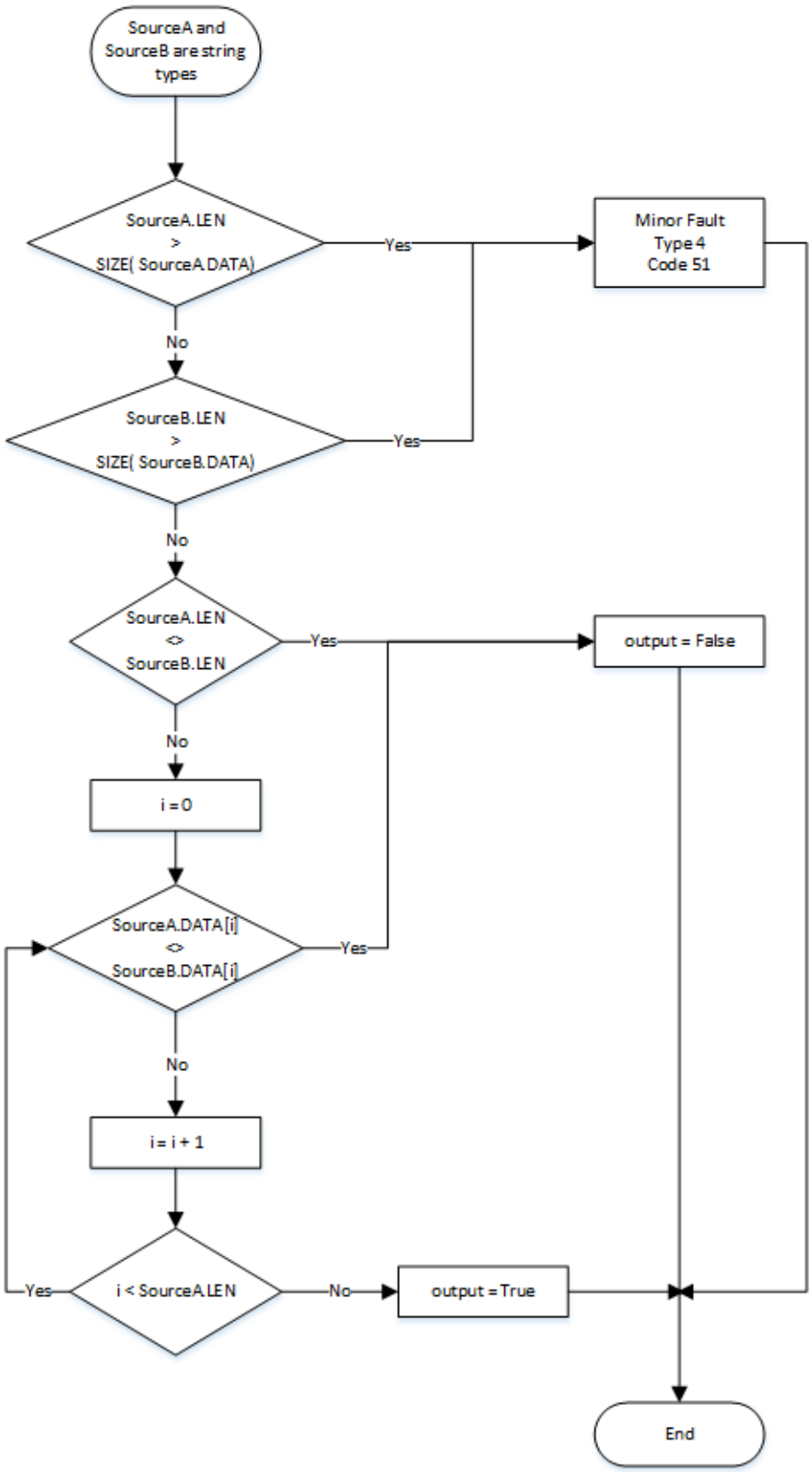
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에 적용됩니다.

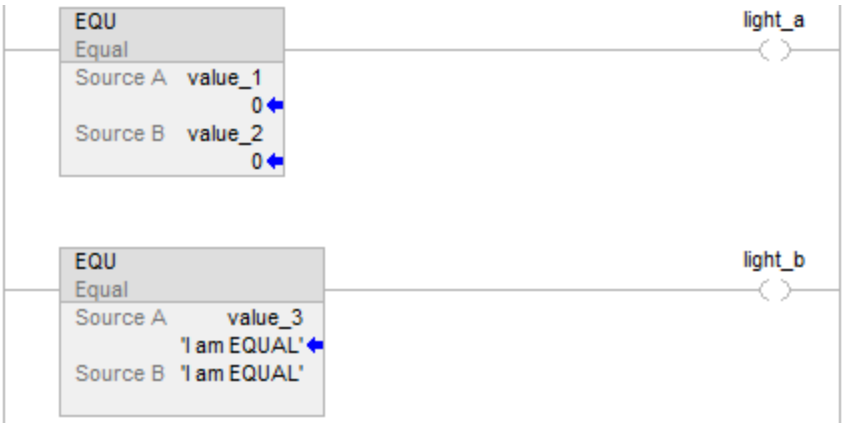
조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	**숫자 비교:** SourceA 와 SourceB 가 NAN 이 아니고 SourceA 와 SourceB 가 같은 경우. Dest 를 참으로 설정 else Dest 를 거짓으로 해제
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

EQU 문자열 비교 흐름도



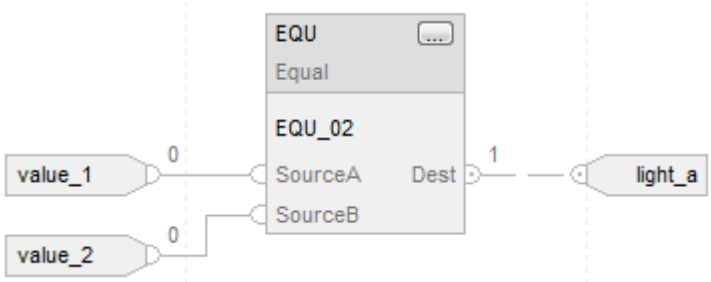
예제

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 평션

ST(스트럭쳐드 텍스트)

```
if value_1 = value_2 then
  light_a := 1;
else
  light_a := 0;
end_if;
```

```
if value_3 = 'I am EQUAL' then
  light_b := 1;
else
  light_b := 0;
end_if;
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

배열을 통한 인덱스 페이지의 978

즉시 값 페이지의 967

FBD 펑션 페이지의 472

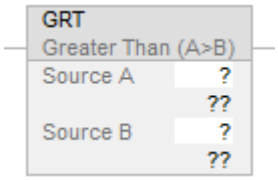
큼(GRT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, GRT 명령어와 > 연산자는 소스 A 가 소스 B 보다 큰지 테스트합니다.

사용 가능한 언어

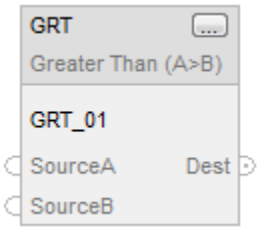
래더 다이어그램



함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 > 연산자를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

숫자 비교

피연산자	데이터 유형 **CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러**	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	형식	설명
Source A	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source B 와 비교해 테스트할 값
Source B	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source A 와 비교해 테스트할 값

문자열 비교

팁: 즉시 문자열 리터럴은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에만 적용됩니다.

피연산자	데이터 유형	형식	설명
Source A	문자열 유형	즉시 리터럴 값 태그	Source B 와 비교해 테스트할 문자열

Source B	문자열 유형	즉시 리터럴 값 태그	Source A 와 비교해 테스트할 문자열

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
GRT	FBD_COMPARE	태그	GRT 구조

FBD_COMPARE 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	REAL	SourceB 와 비교해 테스트할 값
SourceB	REAL	SourceA 와 비교해 테스트할 값

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었는지를 나타냅니다.
Dest	BOOL	SourceA 가 SourceB 보다 큰 경우 참으로 설정합니다. SourceA 가 SourceB 보다 크지 않으면 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	설명
SourceA(상위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceB 와 비교해 테스트할 값
SourceB(하위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceA 와 비교해 테스트할 값

출력 피연산자(오른쪽 핀)	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	설명
Dest	BOOL	SourceA 가 SourceB 보다 큰 경우 참으로 설정합니다. SourceA 가 SourceB 보다 크지 않으면 거짓으로 해제됩니다.

*FBD 평션*을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

폴트에 대해서는 *GRT 문자열 비교 흐름도*를 참조하십시오.

배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	**숫자 비교:** Source A 및 Source B 가 NAN 이 아니고 Source A 가 Source B 보다 큰 경우. 렁-출력-조건을 참으로 설정 else 렁-출력-조건을 거짓으로 해제
	문자열 비교: *GRT 문자열 비교 흐름도*를 참조하십시오. 출력이 거짓인 경우 렁-출력-조건을 거짓으로 해제 그렇지 않으면 렁-출력-조건을 참으로 설정
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

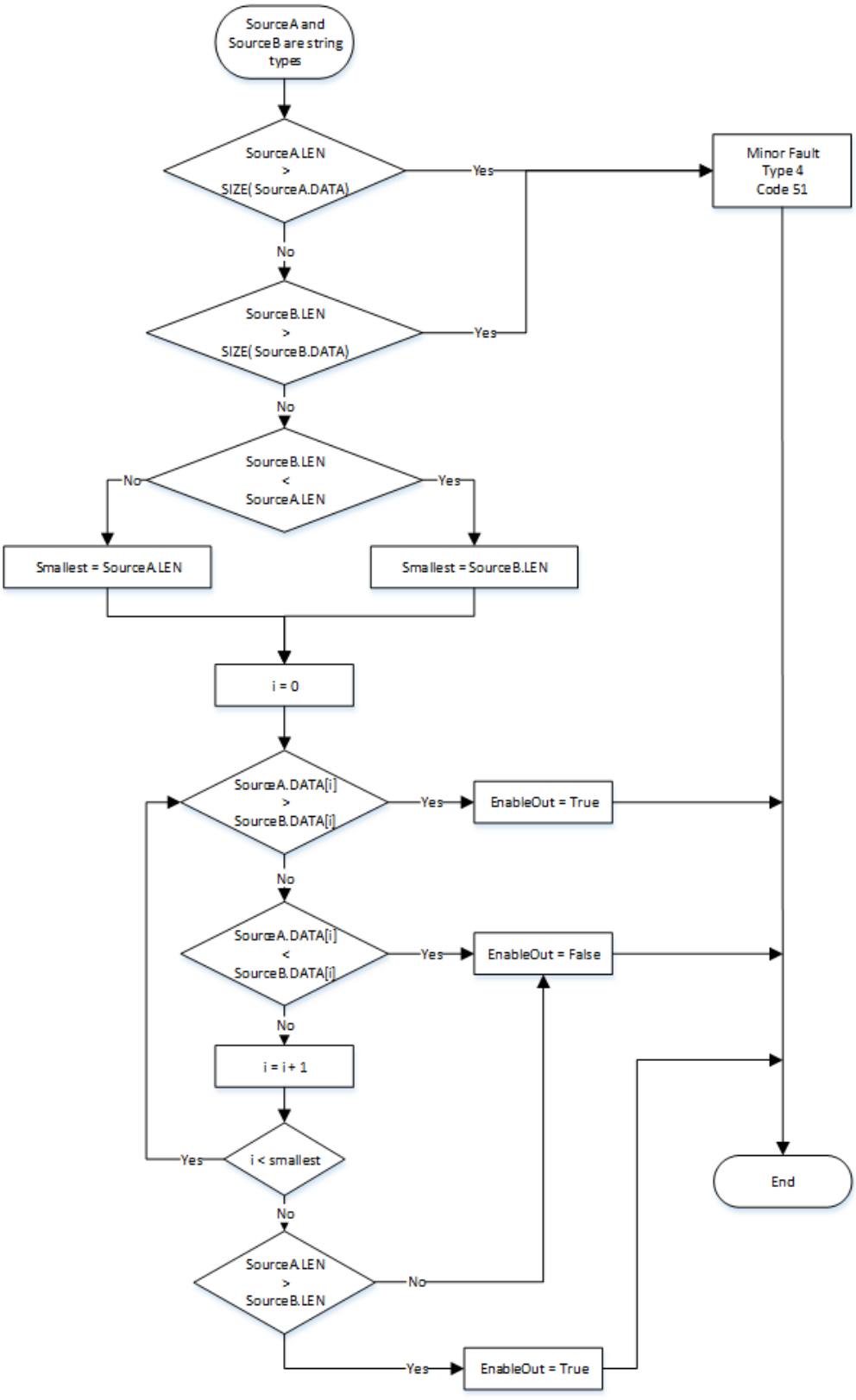
조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	**숫자 비교:** EnableOut 에서 EnableIn 으로 설정 SourceA 와 SourceB 가 NAN 이 아니고 SourceA 가 SourceB 보다 큰 경우. Dest 를 참으로 설정 else Dest 를 거짓으로 해제
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

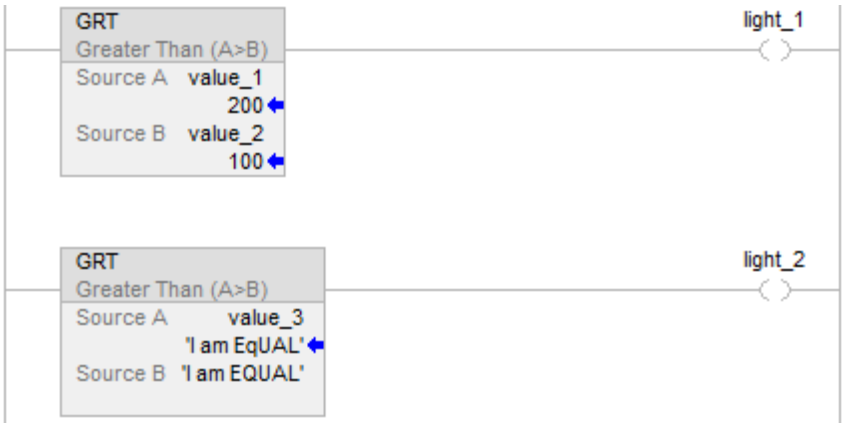
조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	**숫자 비교:** SourceA 와 SourceB 가 NAN 이 아니고 SourceA 가 SourceB 보다 큰 경우. Dest 를 참으로 설정 else Dest 를 거짓으로 해제
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

GRT 문자열 비교 흐름도



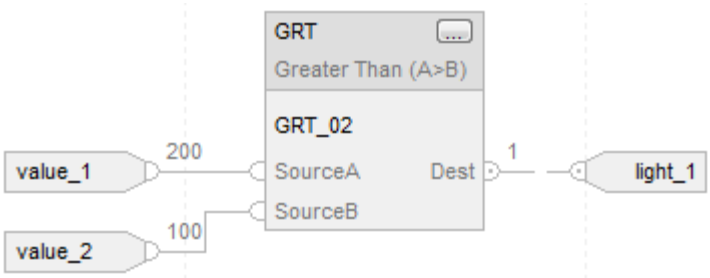
예

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 평션

ST(스트럭처드 텍스트)

```
if value_1 > value_2 then
  light_1 := 1;
else
  light_1 := 0;
end_if;
```

```
if value_3 > 'I am EQUAL' then
  light_2 := 1;
else
  light_2 := 0;
end_if;
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

배열을 통한 인덱스 페이지의 978

즉시 값 페이지의 967

FBD 펑션 페이지의 472

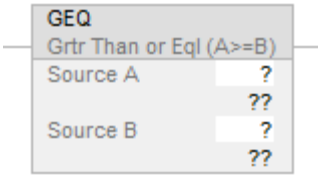
크거나 같음(GEQ)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, GEQ 명령어와 >= 연산자는 소스 A 가 소스 B 보다 크거나 같은지 테스트합니다.

사용 가능한 언어

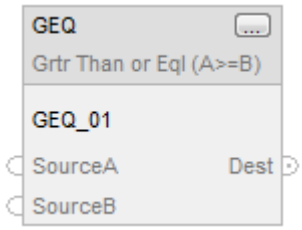
래더 다이어그램



함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 `>=` 연산자를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

숫자 비교

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
Source A	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source B 와 비교해 테스트할 값
Source B	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source A 와 비교해 테스트할 값

문자열 비교

팁: 즉시 문자열 리터럴은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에만 적용됩니다.

피연산자	데이터 유형	형식	설명
Source A	문자열 유형	즉시 리터럴 값 태그	Source B 와 비교해 테스트할 문자열
Source B	문자열 유형	즉시 리터럴 값 태그	Source A 와 비교해 테스트할 문자열

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
GEQ	FBD_COMPARE	태그	GEQ 구조

FBD_COMPARE 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	REAL	SourceB 와 비교해 테스트할 값
SourceB	REAL	SourceA 와 비교해 테스트할 값

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었는지를 나타냅니다.
Dest	BOOL	SourceA 가 SourceB 보다 크거나 같은 경우 참으로 설정합니다. SourceA 가 SourceB 미만인 경우 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형	설명
SourceA(상위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceB 와 대조하여 검사할 값.
SourceB(하위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceA 와 대조하여 검사할 값.

출력 피연산자(오른쪽 핀)	데이터 유형	설명
Dest	BOOL	SourceA 가 SourceB 보다 크거나 같은 경우 참으로 설정합니다. SourceA 가 SourceB 미만인 경우 거짓으로 해제됩니다.

*FBD 평션*을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

폴트에 대해서는 *GEQ 문자열 비교 흐름도*를 참조하십시오.

배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	**숫자 비교:** Source A 및 Source B 가 NAN 이 아니고 Source A 가 Source B 보다 크거나 같은 경우. 렁-출력-조건을 참으로 설정 else 렁-출력-조건을 거짓으로 해제
	문자열 비교: GEQ 문자열 비교 흐름도를 참조하십시오. 출력이 거짓인 경우 렁-출력-조건을 거짓으로 해제 else 렁-출력-조건을 참으로 설정
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	**숫자 비교:** EnableOut 에서 EnableIn 으로 설정 SourceA 와 SourceB 가 NAN 이 아니고 SourceA 가 SourceB 보다 크거나 같은 경우. Dest 를 참으로 설정 else Dest 를 거짓으로 해제
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

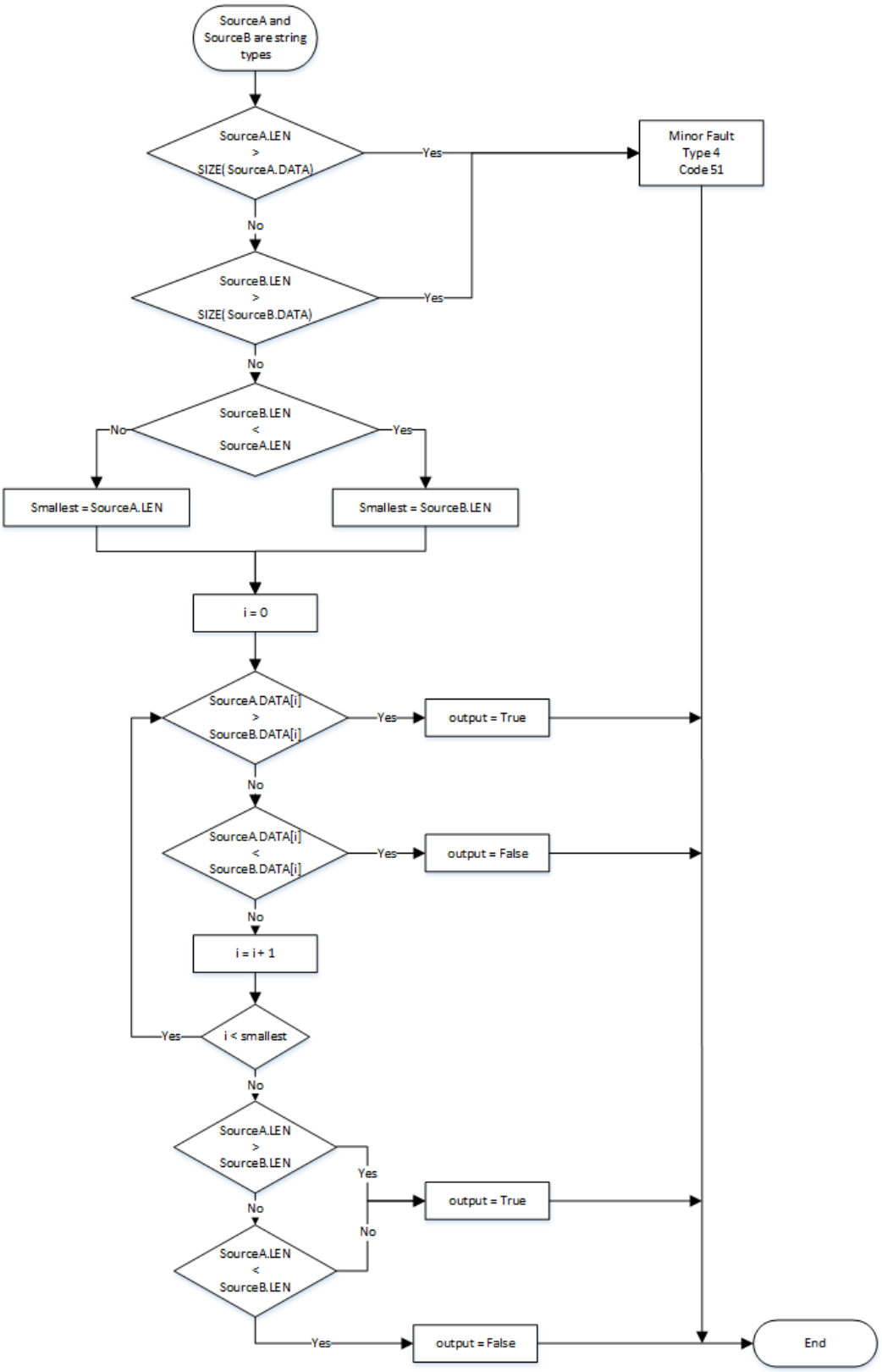
팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A

정상 스캔	**숫자 비교:**
	SourceA 와 SourceB 가 NAN 이 아니고 SourceA 가 SourceB 보다 크거나 같은 경우.
	  Dest 를 참으로 설정
	else
	  Dest 를 거짓으로 해제

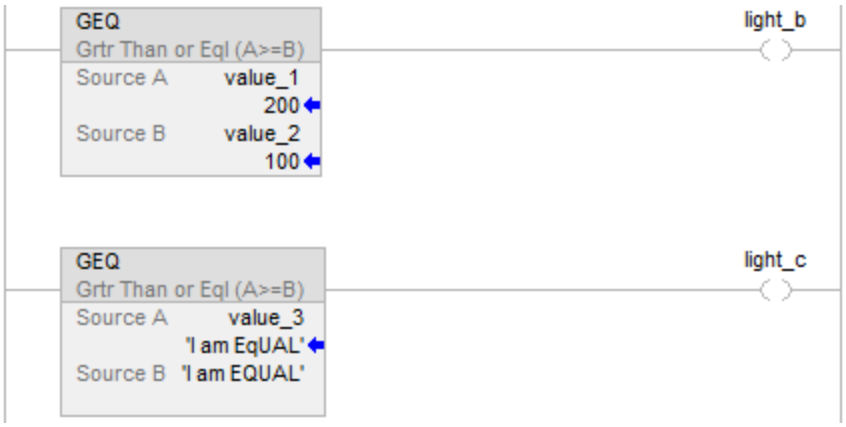
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

GEQ 문자열 비교 흐름도



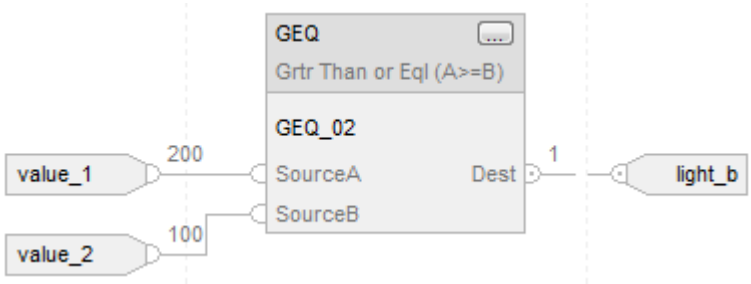
예

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 평션

ST(스트럭쳐드 텍스트)

```
if value_1 >= value_2 then
        light_b := 1;


else
        light_b := 0;
```

```
end_if;

if value_3 >= 'I am EQUAL' then
    light_c := 1;
else
    light_c := 0;
end_if;
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

배열을 통한 인덱스 페이지의 978

즉시 값 페이지의 967

FBD 펑션 페이지의 472

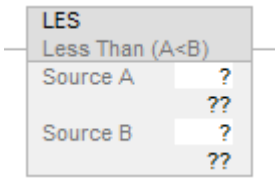
작음(LES)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, LES 명령어와 < 연산자는 소스 A 가 소스 B 보다 작은지 테스트합니다.

사용 가능한 언어

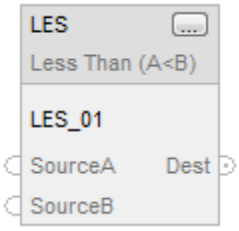
래더 다이어그램



함수 블록 다이어그램

평션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 < 연산자를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

숫자 비교

피연산자	데이터 유형 **CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러**	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	형식	설명
Source A	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source B 와 비교해 테스트할 값
Source B	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source A 와 비교해 테스트할 값

문자열 비교

팁: 즉시 문자열 리터럴은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에만 적용됩니다.

피연산자	데이터 유형	형식	설명
Source A	문자열 유형	즉시 리터럴 값 태그	Source B 와 비교해 테스트할 문자열
Source B	문자열 유형	즉시 리터럴 값 태그	Source A 와 비교해 테스트할 문자열

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
LES	FBD_COMPARE	태그	LES 구조

FBD_COMPARE 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	REAL	SourceB 와 비교해 테스트할 값
SourceB	REAL	SourceA 와 비교해 테스트할 값

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었는지를 나타냅니다.
Dest	BOOL	SourceA 가 SourceB 미만인 경우 참으로 설정합니다. SourceA 가 SourceB 보다 작지 않으면 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	설명
SourceA(상위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceB 와 대조하여 검사할 값.
SourceB(하위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceA 와 대조하여 검사할 값.

출력 피연산자(오른쪽 핀)	데이터 유형	설명
Dest	BOOL	SourceA 가 SourceB 미만인 경우 참으로 설정합니다. SourceA 가 SourceB 보다 작지 않으면 거짓으로 해제됩니다.

*FBD 평션*을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

폴트에 대해서는 *LES 문자열 비교 흐름도*를 참조하십시오.

배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	**숫자 비교:** Source A 및 Source B 가 NAN 이 아니고 Source A 가 Source B 미만인 경우. 렁-출력-조건을 참으로 설정 else 렁-출력-조건을 거짓으로 해제
	문자열 비교: LES 문자열 비교 흐름도를 참조하십시오. 출력이 거짓인 경우 렁-출력-조건을 거짓으로 해제 else 렁-출력-조건을 참으로 설정
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

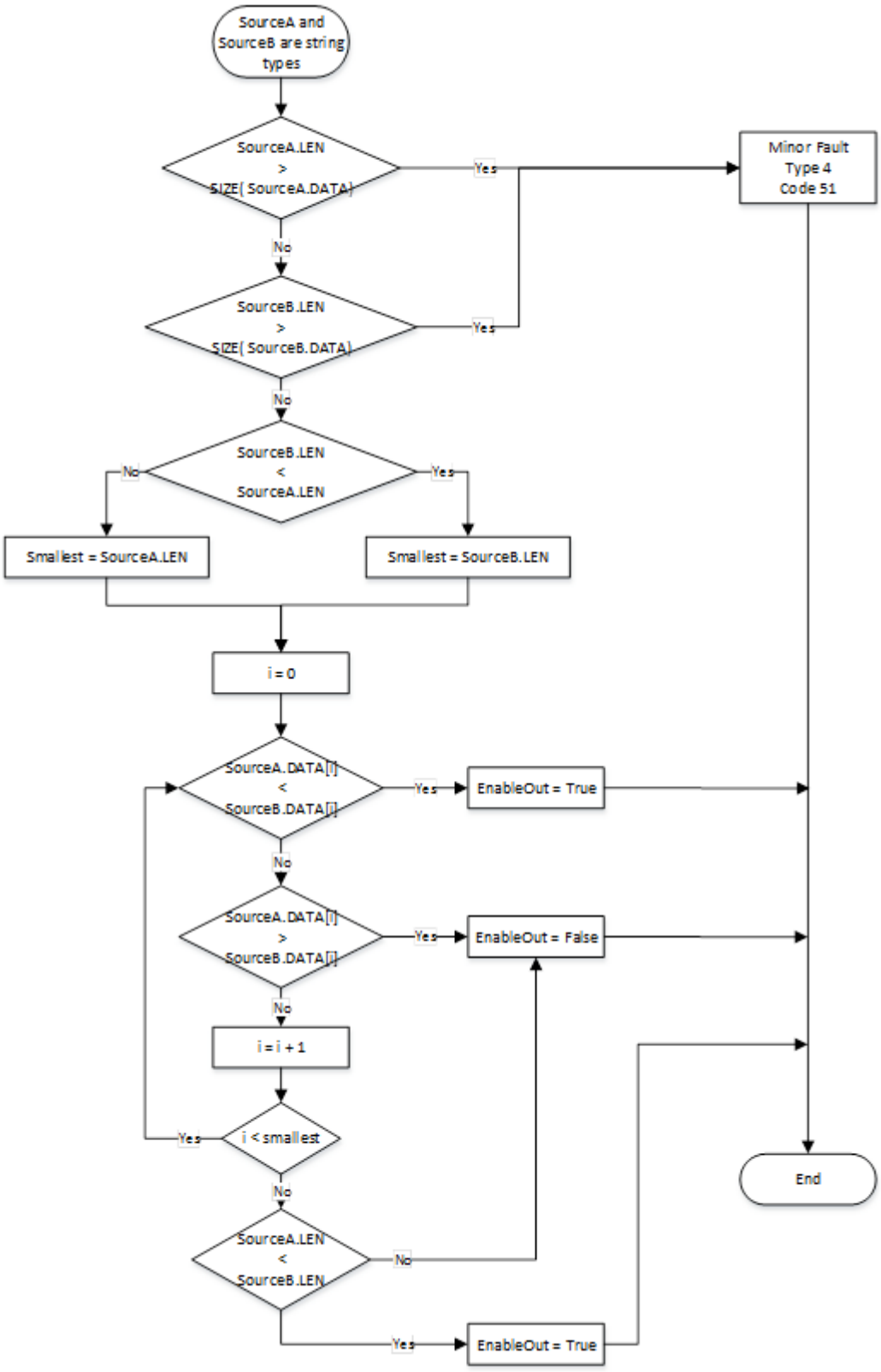
조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	**숫자 비교:** SourceA 와 SourceB 가 NAN 이 아니고 SourceA 가 SourceB 미만인 경우. Dest 를 참으로 설정 else Dest 를 거짓으로 해제
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

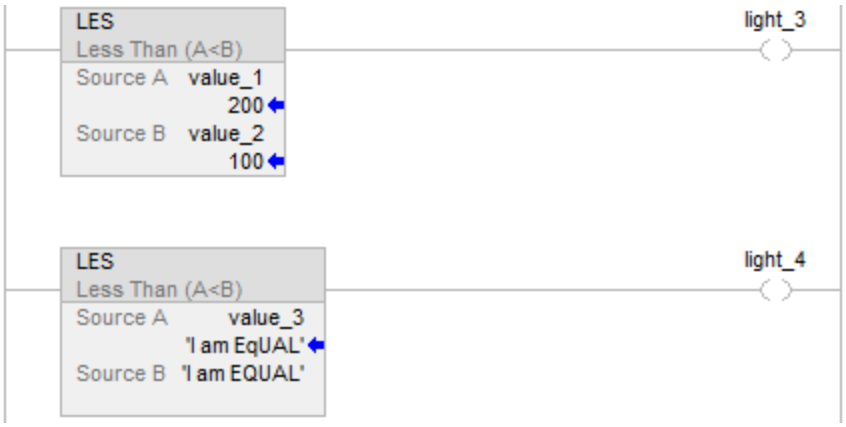
조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	**숫자 비교:** EnableOut 에서 EnableIn 으로 설정 SourceA 와 SourceB 가 NAN 이 아니고 SourceA 가 SourceB 미만인 경우. Dest 를 참으로 설정 else Dest 를 거짓으로 해제
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

LES 문자열 비교 흐름도



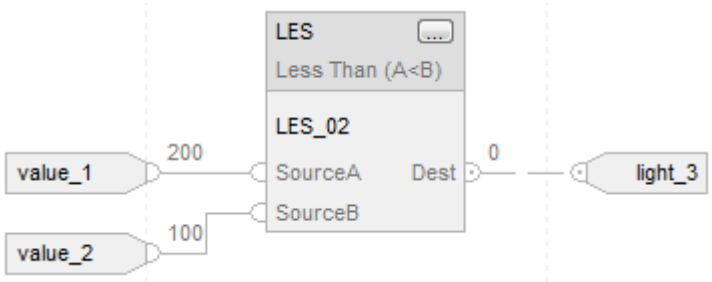
예

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 평션

ST(스트럭처드 텍스트)

```
if value_1 < value_2 then
    light_3 := 1;
else
    light_3 := 0;
end_if;
```

```
if value_3 < 'I am EQUAL' then
    light_4 := 1;
else
    light_4 := 0;
end_if;
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

배열을 통한 인덱스 페이지의 978

즉시 값 페이지의 967

FBD 펑션 페이지의 472

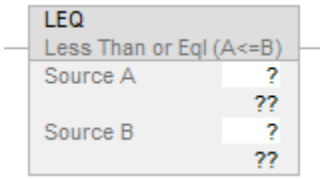
작거나 같음(LEQ)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, LEQ 명령어와 <= 연산자는 소스 A 가 소스 B 보다 작거나 같은지 테스트합니다.

사용 가능한 언어

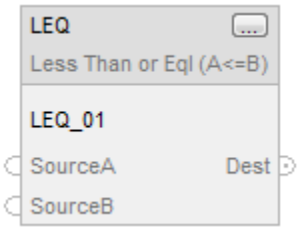
래더 다이어그램



함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 <= 연산자를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

숫자 비교

피연산자	데이터 유형 **CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러**	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	형식	설명
Source A	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source B 와 비교해 테스트할 값
Source B	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source A 와 비교해 테스트할 값

문자열 비교

팁: 즉시 문자열 리터럴은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에만 적용됩니다.

피연산자	데이터 유형	형식	설명
Source A	문자열 유형	즉시 리터럴 값 태그	Source B 와 비교해 테스트할 문자열
Source B	문자열 유형	즉시 리터럴 값 태그	Source A 와 비교해 테스트할 문자열

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
LEQ	FBD_COMPARE	태그	LEQ 구조

FBD_COMPARE 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	REAL	SourceB 와 비교해 테스트할 값
SourceB	REAL	SourceA 와 비교해 테스트할 값

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었는지를 나타냅니다.
Dest	BOOL	SourceA 가 SourceB 보다 작거나 같은 경우 참으로 설정합니다. SourceA 가 SourceB 보다 크면 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	설명
SourceA(상위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceB 와 대조하여 검사할 값.
SourceB(하위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceA 와 대조하여 검사할 값.

출력 피연산자(오른쪽 핀)	데이터 유형	설명
Dest	BOOL	SourceA 가 SourceB 보다 작거나 같은 경우 참으로 설정합니다. SourceA 가 SourceB 보다 크면 거짓으로 해제됩니다.

*FBD 평션*을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

폴트에 대해서는 *LEQ 문자열 비교 흐름도*를 참조하십시오.

배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	**숫자 비교:** Source A 및 Source B가 NAN이 아니고 Source A가 Source B보다 작거나 같은 경우. 렁-출력-조건을 참으로 설정 else 렁-출력-조건을 거짓으로 해제
	문자열 비교: *LEQ 문자열 비교 흐름도*를 참조하십시오. 출력이 거짓인 경우 렁-출력-조건을 거짓으로 해제 else 렁-출력-조건을 참으로 설정
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

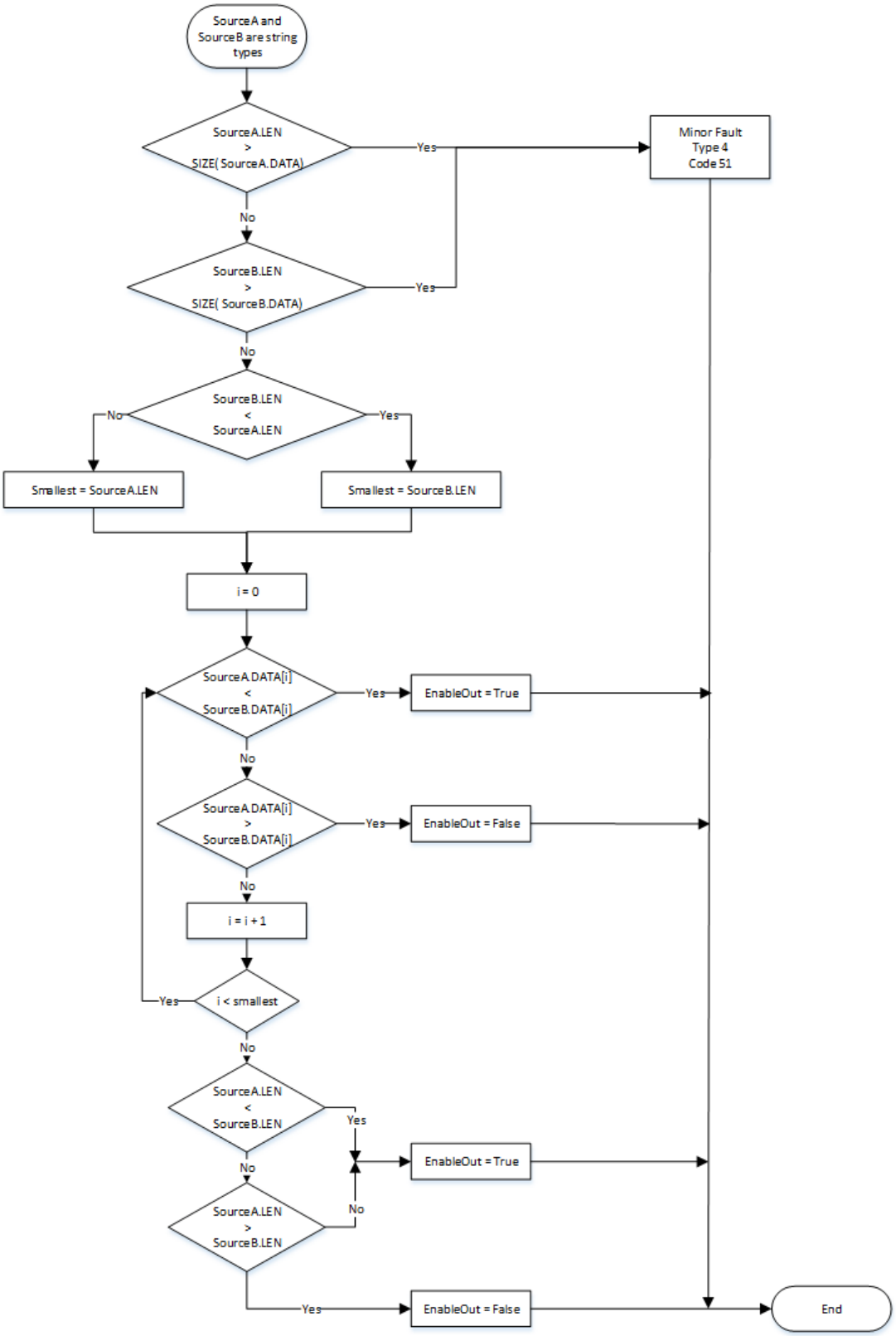
조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	**숫자 비교:** EnableOut 에서 EnableIn 으로 설정 SourceA 와 SourceB 가 NAN 이 아니고 SourceA 가 SourceB 보다 작거나 같은 경우. Dest 를 참으로 설정 else Dest 를 거짓으로 해제
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에 적용됩니다.

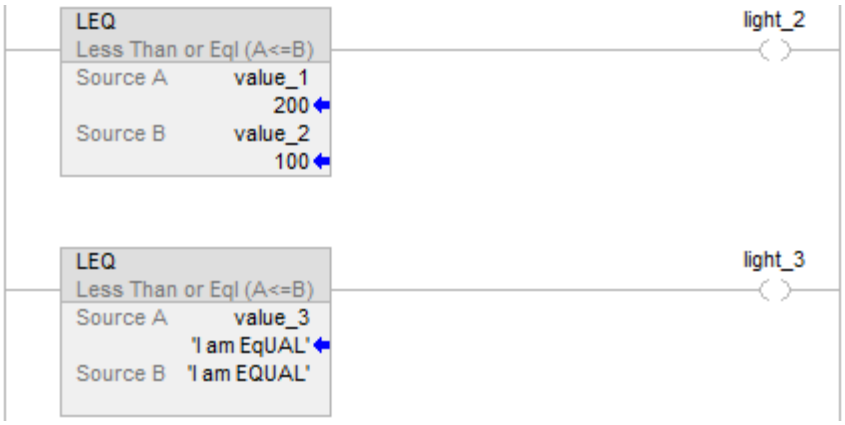
조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	**숫자 비교:** SourceA 와 SourceB 가 NAN 이 아니고 SourceA 가 SourceB 보다 작거나 같은 경우. Dest 를 참으로 설정 else Dest 를 거짓으로 해제
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

LEQ 문자열 비교 흐름도



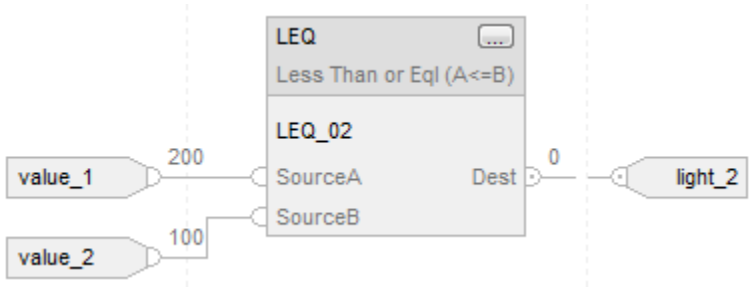
예

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 평션

ST(스트럭처드 텍스트)

```
if value_1 <= value_2 then
        light_2 := 1;
else
        light_2 := 0;
```

```
end_if;

if value_3 <= 'I am EQUAL' then

    light_3 := 1;

else

    light_3 := 0;

end_if;
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

배열을 통한 인덱스 페이지의 978

즉시 값 페이지의 967

FBD 펑션 페이지의 472

제한(LIM)

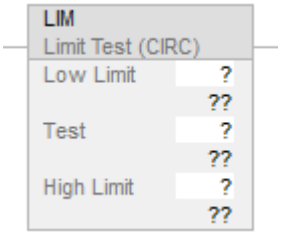
이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

LIM 명령어는 테스트 값이 LIM 흐름도(참)에 표시된 하한 및 상한 범위 이내이거나 범위를 벗어나는지 테스트합니다.

숫자가 아닌(NAN) 피연산자가 있는 경우 .EnableOut 이 거짓으로 해제됩니다.

사용 가능한 언어

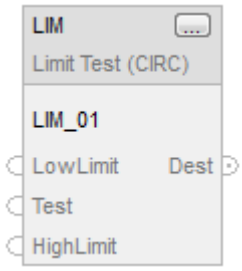
래더 다이어그램



함수 블록 다이어그램

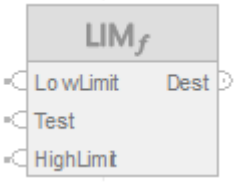
평션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 평션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.



ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
하한(Low Limit)	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	하한 값.
테스트(Test)	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	제한값과 대조하여 검사할 값.
상한(High Limit)	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	상한 값

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
LIM	FBD_LIMIT	태그	LIM 구조

FBD_LIMIT 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
LowLimit	REAL	하한 값.
테스트(Test)	REAL	제한값과 대조하여 검사할 값.
HighLimit	REAL	상한 값

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었는지를 나타냅니다.
Dest	BOOL	Limit test가 참인 경우 참으로 설정되고, Limit test 가 거짓인 경우 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

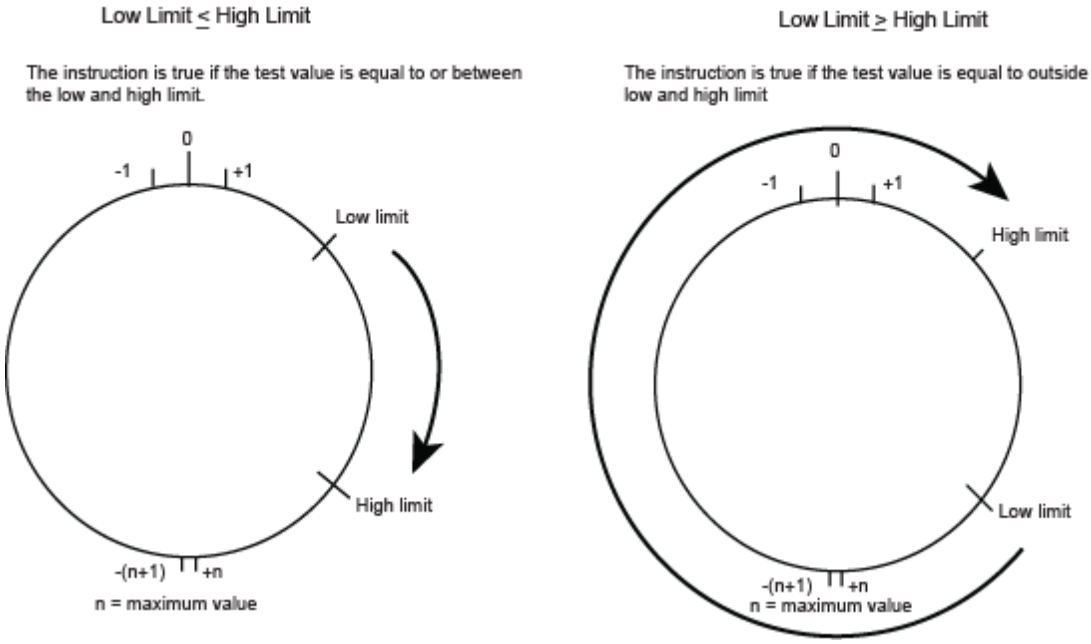
입력 피연산자(왼쪽 핀)	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	설명
하한(Low Limit)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	하한 값
테스트(Test)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	제한값과 대조하여 검사할 값.
상한(High Limit)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	상한 값

출력 피연산자(오른쪽 핀)	데이터 유형	설명
Dest	BOOL	Limit test 가 참인 경우 참으로 설정되고, Limit test 가 거짓인 경우 거짓으로 해제됩니다.

*FBD 평션*을 참조하십시오.

연산

이 섹션에서는 LIM 명령어에 대한 작업을 보여줍니다.



Low Limit 이 다음에 해당하는 경우	그리고 테스트 값이 다음에 해당하는 경우	그에 따른 EnableOut
< 또는 = High Limit(또 이하)	제한과 같거나 제한 이내 제한과 같지 않거나 제한을 벗어남	참임 거짓임
> High Limit	제한과 같거나 제한을 벗어남 제한과 같지 않거나 제한 범위 이내	참임 거짓임

최상위 비트가 참인 경우 부호가 있는 정수는 양의 최대 숫자에서 음의 최대 숫자로 전환됩니다. 예를 들어 16 비트 정수(INT 유형)에서 양의 최대 정수는 32,767 으로 16 진수 단위로는 16#7FFF(0 ~ 14 비트가 모두 참임)로 표시됩니다. 숫자를 1 씩 증가시키면 결과는 16#8000(비트 15 가 참임)입니다. 부호가 있는 정수의 경우 16 진수인 16#8000 은 10 진수 단위로는 -32,768 과

같습니다. 이 포인트로부터 16 비트 모두 설정될 때까지 숫자를 늘리면 결국 16#FFFF 로 되고 이 숫자는 10 진수 -1 과 같습니다.

이 과정은 순환 수직선으로 표시됩니다. LIM 명령어는 하한에서 시작하여 상한에 도달할 때까지 시계 방향으로 증가합니다. 하한에서 상한까지 시계 방향 범위 이내의 모든 Test 값은 EnableOut 을 참으로 설정합니다. 상한에서 하한까지 시계 방향 범위 이내의 모든 테스트 값은 EnableOut 을 거짓으로 해제합니다.

숫자가 아닌(NAN) 피연산자가 있는 경우 .EnableOut 이 거짓으로 해제됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	*LIM 흐름도(참)*를 참조하십시오. 출력이 참인 경우 렁-출력-조건을 참으로 설정합니다. else 렁-출력-조건을 거짓으로 해제
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

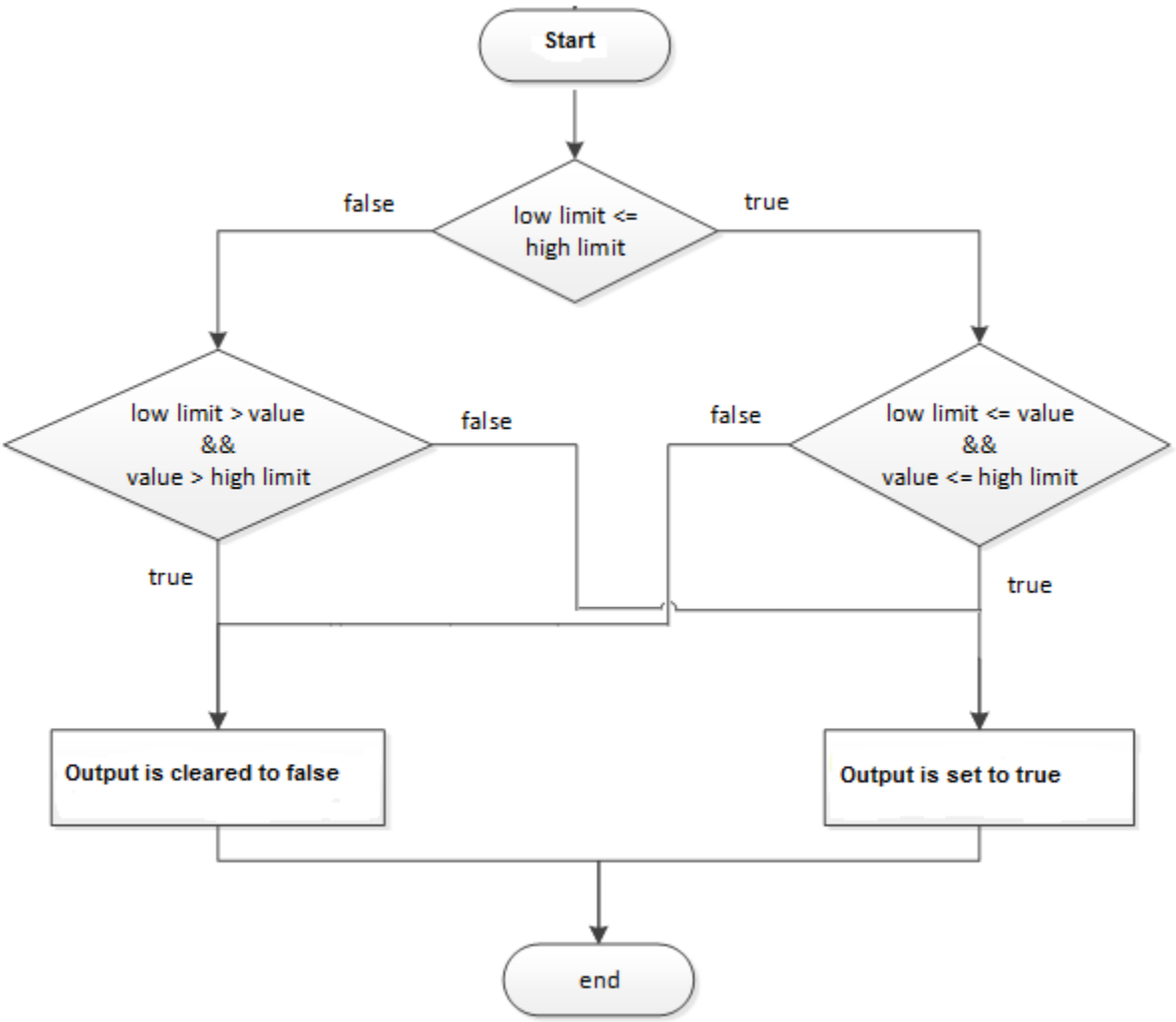
조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 을 EnableIn 으로 설정합니다.
EnableIn 이 참임	EnableOut 을 EnableIn 으로 설정합니다. See LIM 흐름도(참) 참조 Dest = 출력값
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	LIM 흐름도(참)를 참조하십시오. Dest = 출력
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

LIM 흐름도(참)

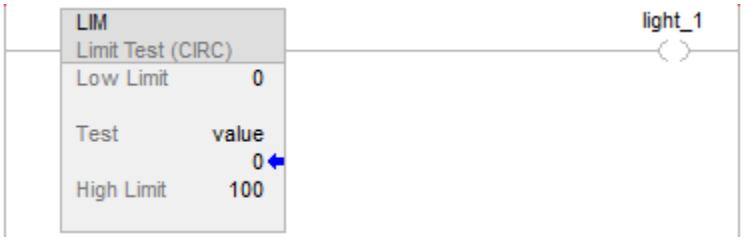


예제

예 1: Low Limit <= High Limit

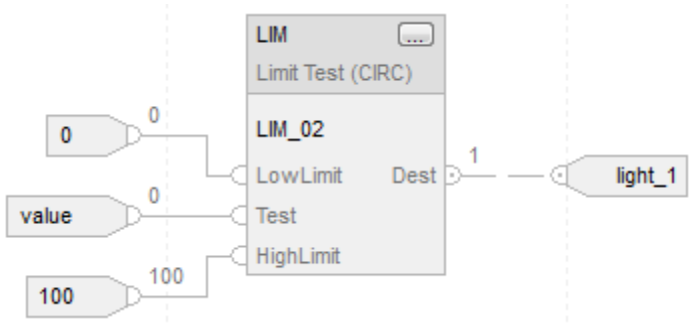
테스트 값이 하한 이상일 때와 테스트 값이 상한 이하일 때 light_1 이 설정됩니다.

래더 다이어그램

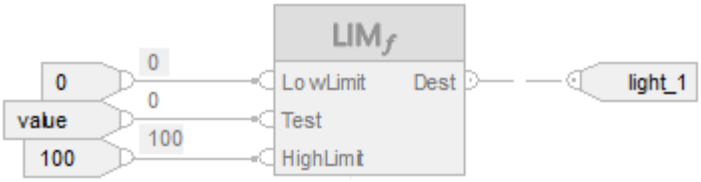


함수 블록 다이어그램

FBD 블록



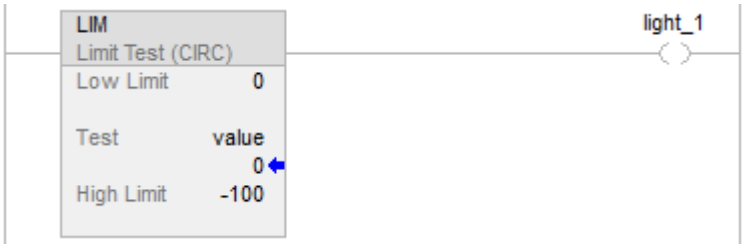
FBD 평션



예 2: Low Limit > High Limit

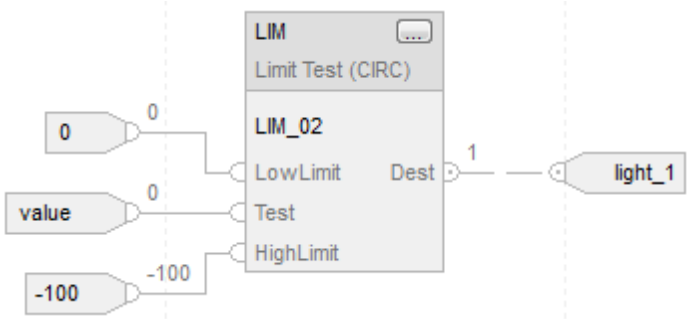
value > 또는 = 0 이거나 value < 또는 = -100 인 경우 light_1 을 참으로 설정하고, value < 0 또는 value > -100 인 경우 light_1 을 거짓으로 해제합니다.

래더 다이어그램

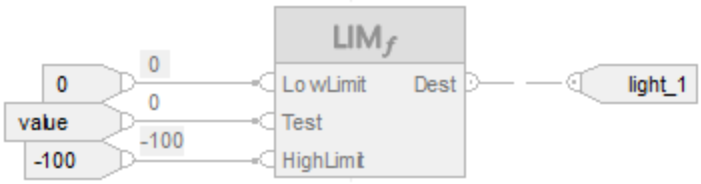


함수 블록 다이어그램

FBD 블록



FBD 펑션



추가 참조

비교 명령어 페이지의 329

데이터 변환 페이지의 967

배열을 통한 인덱스 페이지의 978

즉시 값 페이지의 967

FBD 펑션 페이지의 472

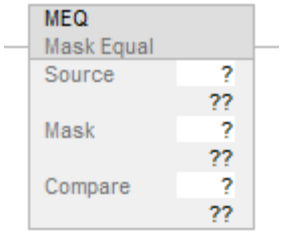
마스크 같음(MEQ)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

MEQ 명령어는 마스크를 통해 Source 및 Compare 값을 전달하고 결과를 비교합니다.

사용 가능한 언어

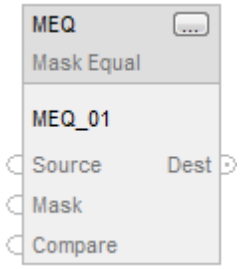
래더 다이어그램



함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
소스	SINT INT DINT	SINT INT DINT LINT USINT UINT UDINT ULINT	즉시 태그	비교 값과 대조하여 검사할 값.
마스크(Mask)	SINT INT DINT	SINT INT DINT LINT USINT UINT UDINT ULINT	즉시 태그	차단 또는 통과시킬 비트.
비교(Compare)	SINT INT DINT	SINT INT DINT LINT USINT UINT UDINT ULINT	즉시 태그	소스와 대조하여 검사할 값.

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
MEQ	FBD_MASK_EQUAL	태그	MEQ 구조

FBD_MASK_EQUAL 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
소스	DINT	비교 값과 대조하여 검사할 값.
마스크(Mask)	DINT	차단할 비트를 정의합니다(예: 마스크).
비교(Compare)	DINT	소스와 대조하여 검사할 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	BOOL	결과가 참인 경우 참으로 설정되고, 결과가 거짓인 경우 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	설명
소스	SINT INT DINT LINT USINT UINT UDINT ULINT	비교 값과 대조하여 검사할 값.
마스크(Mask)	SINT INT DINT LINT USINT UINT UDINT ULINT	차단 또는 통과시킬 비트.
비교(Compare)	SINT INT DINT LINT USINT UINT UDINT ULINT	소스와 대조하여 검사할 값.
	SINT 또는 INT 태그는 영 채우기를 통해 DINT 값으로 변환됩니다.	

출력 피연산자(오른쪽 핀)	데이터 유형	설명
Dest	BOOL	결과가 참인 경우 참으로 설정되고, 결과가 거짓인 경우 거짓으로 해제됩니다.

FBD 평션을 참조하십시오.

연산

마스크의 "1"은 데이터 비트가 전달되었음을 의미합니다. 마스크의 "0"은 데이터 비트가 차단되었음을 의미합니다. 일반적으로 Source, Mask, Compare 값은 모두 동일한 데이터 유형입니다.

SINT 또는 INT 데이터 유형을 사용하는 경우 명령어에서 그 값의 위쪽 비트를 0 으로 채워서 DINT 데이터 유형과 같은 크기로 됩니다.

즉시 마스크 값 입력

마스크를 입력하면 프로그래밍 소프트웨어에서 기본적으로 10 진수 값으로 설정됩니다. 다른 형식을 사용하여 마스크를 입력하려면 값 앞에 알맞은 접두사를 붙입니다.

접두사	설명
16#	16#0F0F 와 같은 16 진수
8#	8#16 과 같은 8 진수
2#	2#00110011 과 같은 2 진수

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	MEQ 흐름도(참)를 참조하십시오. 출력이 참인 경우 렁-출력-조건을 참으로 설정 else 렁-출력-조건을 거짓으로 해제
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 을 EnableIn 으로 설정합니다.

EnableIn 이 참임	EnableOut 을 EnableIn 으로 설정합니다.
	*MEQ 흐름도(참)*를 참조하십시오.
	출력이 참인 경우
	  Dest 를 참으로 설정
	else
	  Dest 를 거짓으로 해제

명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

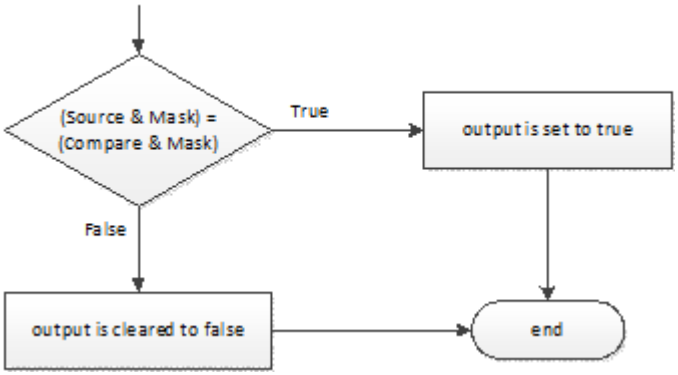
팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A

정상 스캔	*MEQ 흐름도(참)*를 참조하십시오.
	출력이 참인 경우
	  Dest 를 참으로 설정
	else
	  Dest 를 거짓으로 해제

명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

MEQ 흐름도(참)



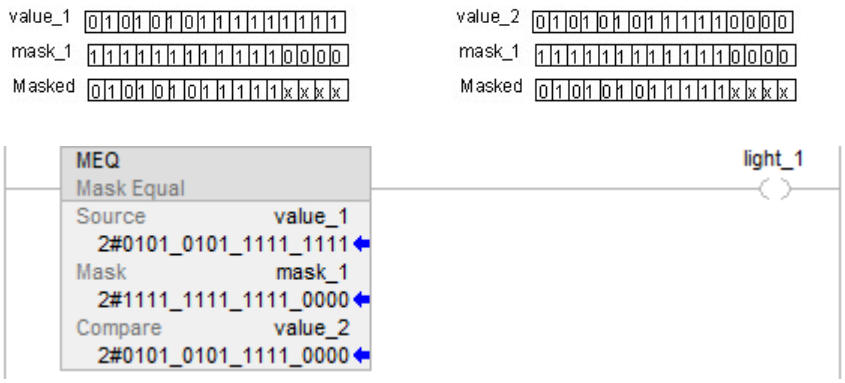
예제

예 1

마스킹된 value_1 이 마스킹된 value_2 와 같으면 light_1 을 참으로 설정합니다. 마스킹된 value_1 이 마스킹된 value_2 와 같지 않으면 light_1 을 거짓으로 해제합니다.

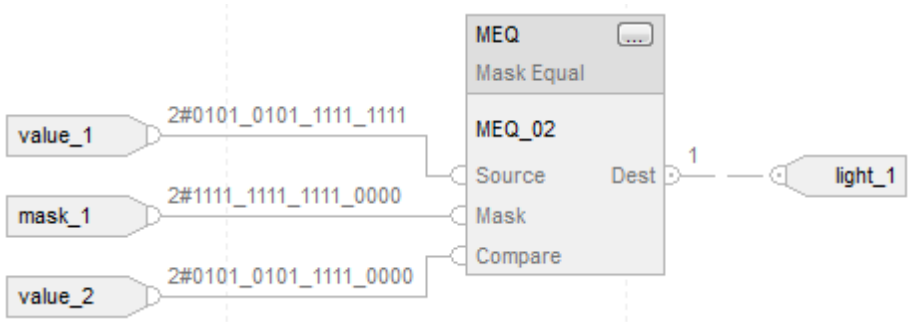
이 예는 마스킹된 값이 같음을 보여줍니다. 마스크의 0 은 명령어가 해당 비트를 비교하지 못하도록 저지합니다(예에서 x 로 표시됨).

래더 다이어그램

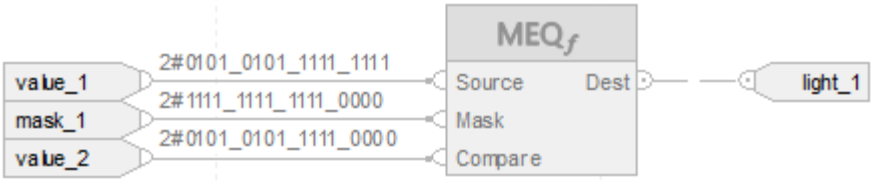


함수 블록 다이어그램

FBD 블록



FBD 평션

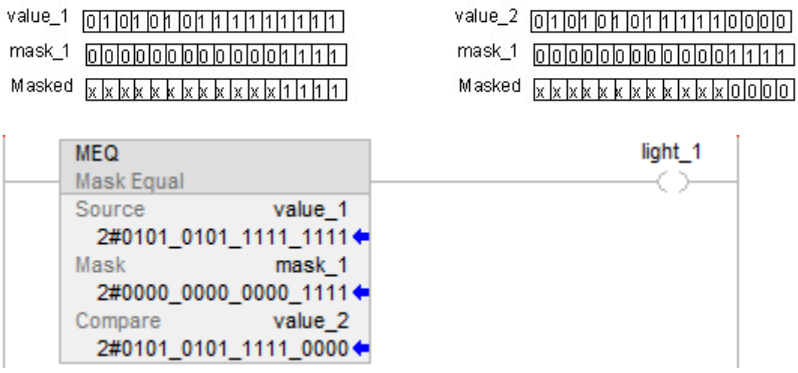


예 2

마스킹된 value_1 이 마스킹된 value_2 와 같으면 light_1 을 참으로 설정합니다. 마스킹된 value_1 이 마스킹된 value_2 와 같지 않으면 light_1 을 거짓으로 해제합니다.

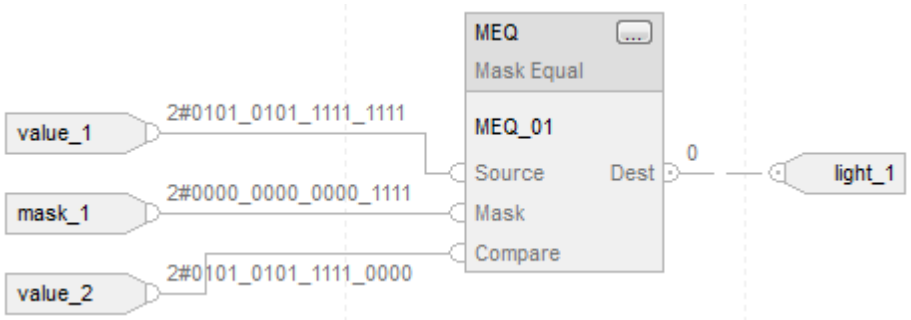
이 예는 마스킹된 값이 같지 않음을 보여줍니다. 마스크의 0 은 명령어가 해당 비트를 비교하지 못하도록 저지합니다(예에서 x 로 표시됨).

래더 다이어그램

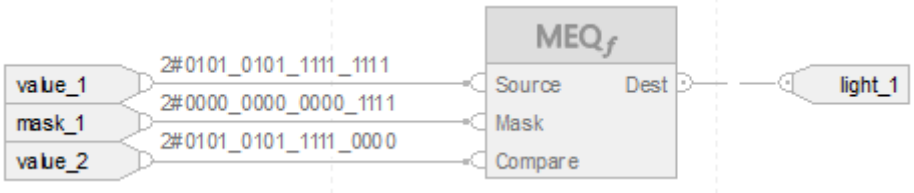


함수 블록 다이어그램

FBD 블록



FBD 평션



추가 참조

배열을 통한 인덱스 페이지의 978

즉시 값 페이지의 967

데이터 변환 페이지의 967

영 채우기란 무엇인가? 페이지의 408

FBD 평션 페이지의 472

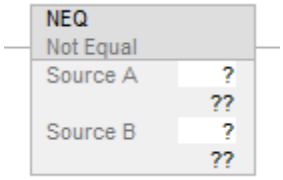
같지 않음(NEQ)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, NEQ 명령어와 <> 연산자는 소스 A 와 소스 B 가 같지 않은지 테스트합니다.

사용 가능한 언어

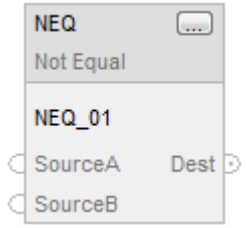
래더 다이어그램



함수 블록 다이어그램

평션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 평션

팁: FBD 평션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 <> 연산자를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

숫자 비교

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	형식	설명
Source A	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source B 와 비교해 테스트할 값
Source B	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source A 와 비교해 테스트할 값

문자열 비교

팁: 즉시 문자열 리터럴은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에만 적용됩니다.

피연산자	데이터 유형	형식	설명
Source A	문자열 유형	즉시 리터럴 값 태그	Source B 와 비교해 테스트할 문자열
Source B	문자열 유형	즉시 리터럴 값 태그	Source A 와 비교해 테스트할 문자열

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
NEQ	FBD_COMPARE	태그	NEQ 구조

FBD_COMPARE 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	REAL	SourceB 와 대조하여 검사할 값.
SourceB	REAL	SourceA 와 대조하여 검사할 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었는지를 나타냅니다.
Dest	BOOL	SourceA 가 SourceB 와 같지 않은 경우 참으로 설정합니다. SourceA 가 SourceB 와 같으면 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형	설명
SourceA(상위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceB 와 비교해 테스트할 값
SourceB(하위)	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	SourceA 와 대조하여 검사할 값.

출력 피연산자(오른쪽 핀)	데이터 유형	설명
Dest	BOOL	SourceA 가 SourceB 와 같지 않은 경우 참으로 설정합니다. SourceA 가 SourceB 와 같으면 거짓으로 해제됩니다.

FBD 펑션을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

폴트에 대해서는 *NEQ 문자열 비교 흐름도*를 참조하십시오.

배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	**숫자 비교:** Source A 또는 Source B 가 NAN 이거나 Source A 가 Source B 와 같지 않은 경우. 렁-출력-조건을 참으로 설정 else 렁-출력-조건을 거짓으로 해제
	문자열 비교: *NEQ 문자열 비교 흐름도*를 참조하십시오. 출력이 거짓인 경우 렁-출력-조건을 거짓으로 해제 그렇지 않으면 렁-출력-조건을 참으로 설정
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

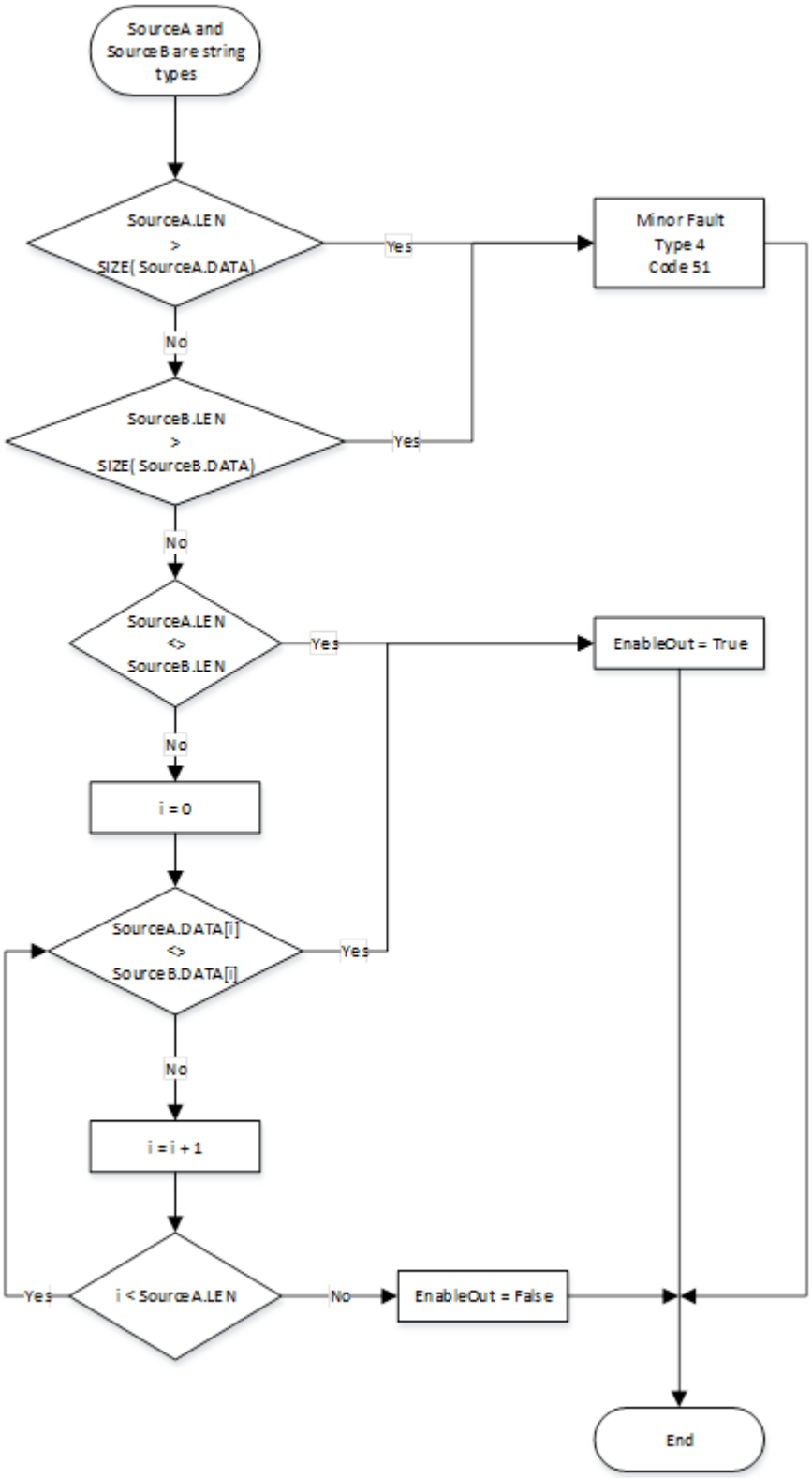
조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	**숫자 비교:** EnableOut 에서 EnableIn 으로 설정 SourceA 또는 SourceB 가 NAN 이거나 SourceA 가 SourceB 와 같지 않은 경우. Dest 를 참으로 설정 else Dest 를 거짓으로 해제
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

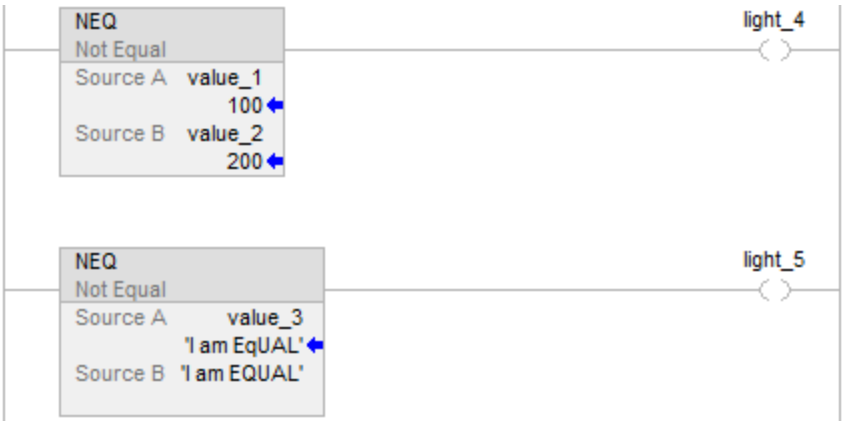
조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	**숫자 비교:** SourceA 또는 SourceB 가 NAN 이거나 SourceA 가 SourceB 와 같지 않은 경우. Dest 를 참으로 설정 else Dest 를 거짓으로 해제
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

NEQ 문자열 비교 흐름도



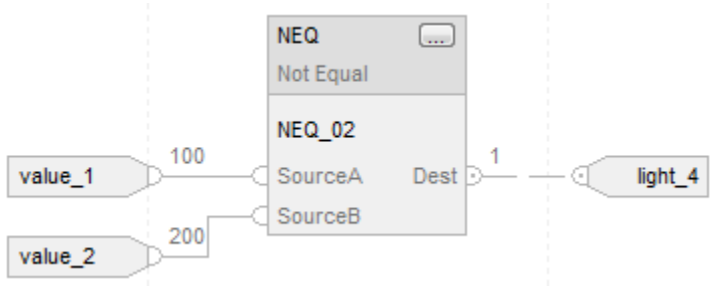
예제

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 평션

ST(스트럭처드 텍스트)

```
if value_1 <> value_2 then
  light_4 := 1;
else
  light_4 := 0;
end_if;
```

```
if value_3 <> 'I am EQUAL' then
  light_5 := 1;
else
  light_5 := 0;
end_if;
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

배열을 통한 인덱스 페이지의 978

즉시 값 페이지의 967

FBD 펑션 페이지의 472

유효한 연산자

다음은 유효한 연신자입니다.

연산자	설명	허용					
		배열 인덱스	FSC	CMP	FAL	CPT	Safety
+	더하기	X	X	X	X	X	X
-	빼기/부정	X	X	X	X	X	X
*	곱하기	X	X	X	X	X	X
/	나누기	X	X	X	X	X	X
=	같음		X	X			X
<	보다 작음		X	X			X
<=	보다 작거나 같음		X	X			X
>	보다 큼		X	X			X
>=	보다 크거나 같음		X	X			X

<>	같지 않음		X	X			X
**	지수(x 의 y 제곱)		X	X	X	X	
ABS	절대값		X	X	X	X	X
ACS	아크 코사인		X	X	X	X	
논리곱	비트별 논리곱	X	X	X	X	X	X
ASN	아크 사인		X	X	X	X	
ATN	아크 탄젠트		X	X	X	X	
COS	코사인		X	X	X	X	
DEG	라디안 -> 도		X	X	X	X	
FRD	BCD -> 정수	X	X	X	X	X	
LN	자연 로그		X	X	X	X	
LOG	로그 밑수 10		X	X	X	X	
MOD	모듈로-나누기		X	X	X	X	X
NOT	비트별 논리부정	X	X	X	X	X	X
또는	비트별 논리합	X	X	X	X	X	X
RAD	도 -> 라디안		X	X	X	X	
SIN	사인		X	X	X	X	
SQR	제곱근	X	X	X	X	X	
TAN	탄젠트		X	X	X	X	
TOD	정수 -> BCD	X	X	X	X	X	
TRN	자르기		X	X	X	X	
XOR	비트별 배타적 논리합	X	X	X	X	X	X

영 채우기란 무엇인가?

작은 정수 유형을 큰 정수 유형으로 변환하는 방법은 다음과 같이 두 가지가 있습니다.

- 영 채우기
- 부호 확장

사용할 방법은 피연산자를 사용하는 명령어에 달려 있습니다.

영 채우기의 경우, 작은 유형의 범위를 초과하는 비트는 모두 0 으로 채웁니다.

예를 들어, SINT: 16#87 = -121 을 DINT 로 변환하면 16#00000087 = 135 가 됩니다.

부호 확장의 경우, 작은 유형의 범위를 초과하는 비트는 모두 작은 유형의 부호 비트로 채웁니다.

예를 들어, SINT: 16#87 = -121 을 DINT 로 변환하면 16#FFFFFF87 = -121 이 됩니다.

추가 참조

마스크 같음(MEQ) 페이지의 389

계산/연산 명령어

계산/연산 명령어

계산/연산 명령어는 식 또는 특정 산술 명령어를 사용하여 산술 연산을 계산합니다.

사용 가능한 명령어

래더 다이어그램

CPT	ADD	SUB	MUL	DIV	MOD	SQR	SQRT	NEG	ABS

함수 블록 다이어그램

FBD 블록

ADD	SUB	MUL	DIV	MOD	SQR	SQRT	NEG	ABS

FBD 평션

$+_f$	$\times_f$	$\div_f$	$\%_f$	$\sqrt{x}_f$	$-x_f$	$\lvert x \rvert_f$
ADD	465	DIV	MOD	SQR/SQRT/	452	ABS

ST(스트럭처드 텍스트)

SQR	SQRT	ABS

실행할 작업:	사용할 명령어:
식 계산	CPT
값 2 개 더하기	ADD
값 2 개 빼기	SUB
값 2 개 곱하기	MUL

값 2 개 나누기	DIV
한 값을 다른 값으로 나눈 후 나머지 확인	MOD
값의 제곱근 계산	SQR
값의 부호가 반대 부호로 바꾸기	NEG
값의 절대값 가져오기	ABS

데이터 유형을 혼합할 수 있지만 정확도 손실 및 반올림 에러가 발생할 수 있으며 명령어를 실행하는 데 시간이 더 걸립니다. 결과가 잘렸는지 확인하려면 S:V 비트를 확인하십시오.

볼드체 데이터 유형은 최적의 데이터 유형을 나타냅니다. 명령어의 모든 피연산자가 동일한 최적의 데이터 유형(일반적으로 DINT 또는 REAL)을 사용하는 경우 명령어는 가장 적은 메모리를 사용하면서 보다 빠르게 실행됩니다.

계산/연산 명령어는 렁-입력-조건이 참인 한 명령어가 스캔될 때마다 한 번씩 실행됩니다. 식을 한 번만 계산하도록 하려면 원샷 명령어를 사용하여 명령어를 트리거하십시오.

추가 참조

비교 명령어 페이지의 329

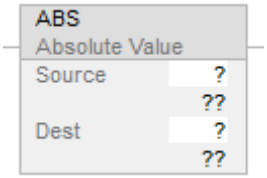
절대값(ABS)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

ABS 명령어와 연산자는 활성화된 경우 Source 의 절대값을 얻습니다. 연산자는 단순히 결과를 반환하는 한편 명령어를 사용하면 Dest 에 결과가 저장됩니다. 결과가 음의 최대 정수값(예: SINT 의 경우 -128, INT 의 경우 -32,768, DINT 의 경우 -2,147,483,648)인 경우 오버플로가 표시됩니다.

사용 가능한 언어

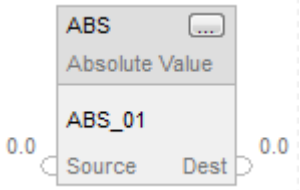
래더 다이어그램



함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 계산하려면 식에서 ABS 를 연산자로 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 ST(스트럭쳐드 텍스트) 구문을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
소스	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	절대값을 얻을 값.
Dest	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	태그	명령어의 결과를 저장하는 태그.

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
ABS	FBD_MATH_ADVANCED	태그	ABS 구조

FBD_MATH_ADVANCED 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
소스	REAL	절대값을 얻을 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	REAL	명령어의 결과.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	설명
소스	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	절대값을 얻을 값.

출력 피연산자(오른쪽 핀)	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	설명
Dest	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	펑션 결과.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. Dest = Source 의 절대값.
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 을 EnableIn 으로 설정합니다.
EnableIn 이 참임	Dest = Source 의 절대값. 오버플로가 발생하는 경우 EnableOut 을 거짓으로 해제합니다. else EnableOut 을 참으로 설정합니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

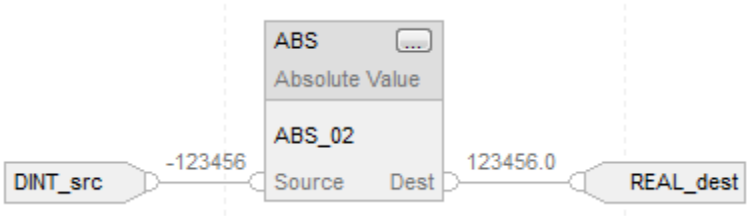
조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	Dest = 소스의 절대값
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

예제

래더 다이어그램

함수 블록 다이어그램

FBD 블록



FBD 펑션

ST(스트럭처드 텍스트)

```
DINT_dest := ABS(DINT_src);
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

즉시 값 페이지의 967

FBD 펑션 페이지의 472

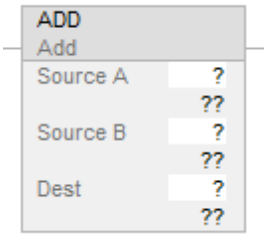
더하기(ADD)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, ADD 명령어와 '+' 연산자는 소스 A 를 소스 B 에 더합니다.

사용 가능한 언어

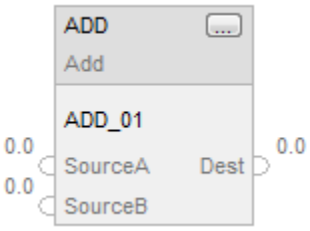
래더 다이어그램



함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션 요소는 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 계산하려면 식에서 '+' 연산자를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	형식	설명
SourceA	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source B 에 더하는 값
SourceB	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	Source A 에 더하는 값
Dest	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	태그	명령어의 결과를 저장하는 태그

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
ADD	FBD_MATH	태그	ADD 구조

FBD_MATH 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	REAL	SourceB 에 더하는 값.
SourceB	REAL	SourceA 에 더하는 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	REAL	명령어의 결과.

FBD 펑션

팁: FBD 펑션 요소는 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러 전용**	설명
SourceA(상위)	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	SourceB 에 더하는 값.

SourceB(하위)	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	SourceA 에 더하는 값.

출력 피연산자 (오른쪽 핀)	**데이터 유형** **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러 전용**	**설명**
Dest	DINT UDINT LINT ULINT REAL LREAL	펑션 결과.

*FBD 펑션*을 참조하십시오.

연산 상태 플래그에 영향

컨트롤러	**연산 상태 플래그에 영향**
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

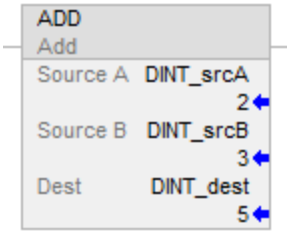
조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정, Dest = Source A + Source B
사후 스캔	N/A

함수 블록 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	Dest = SourceA + SourceB 오버플로가 발생하는 경우 EnableOut 을 거짓으로 해제 else EnableOut 을 참으로 설정
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

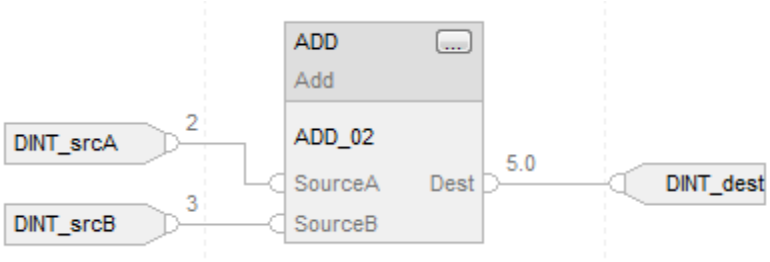
예

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 펑션

ST(스트럭쳐드 텍스트)

```
DINT_dest := DINT_srcA + DINT_srcB;
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

즉시 값 페이지의 967

FBD 펑션 페이지의 472

계산(CPT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

CPT 명령어는 활성화되면 식을 평가하고 Dest 에 결과를 저장합니다.

CPT 명령어는 복잡한 식을 하나의 명령어에 담을 수 있습니다.

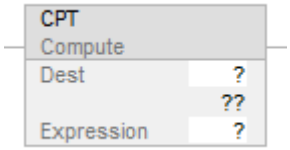
식을 평가할 때 다음 조건 중 하나라도 참인 경우 계산이 수행되기 전에 LREAL 이 아닌 모든 피연산자가 LREAL 로 변환됩니다.

- 식의 모든 피연산자가 LREAL 입니다.
- 이 식에는 SIN, COS, TAN, ASN, ACS, ATN, LN, LOG, DEG, RAD 가 포함됩니다.
- Dest 는 LREAL 입니다.

안전 응용 프로그램에서 허용 가능한 연산자에 대한 규칙이 있습니다. *유효 연산자* 섹션을 참조하십시오.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Dest	SINT INT DINT REAL	태그	결과를 저장하는 태그
Expression	SINT INT DINT REAL	즉시 태그	태그 및/또는 연산자로 구분된 즉시 값으로 구성된 식.

식의 형식

식에 사용된 연산자마다 하나 또는 두 개의 피연산자(태그 또는 즉시 값)를 입력해야 합니다. 다음 표를 사용하여 식 내의 연산자 및 피연산자 형식을 지정합니다.

연산자 작용 대상:	사용 형식:	예
피연산자 1 개	연산자(피연산자)	ABS(tag)
피연산자 2 개	operand_a 연산자 operand_b	tag_b + 5 tag_c AND tag_d (tag_e**2) MOD (tag_f / tag_g)

연산 순서 정의

명령어가 식의 연산을 규정된 순서에 따라 수행합니다. 조건을 괄호 안에 묶어 연산의 순서를 지정합니다. 그러면 명령어가 괄호 안의 연산을 다른 연산보다 먼저 수행합니다.

연산 순서가 같은 경우에는 왼쪽에서 오른쪽 순으로 수행됩니다.

순서	연산
1	( )
2	ABS, ACS, ASN, ATN, COS, DEG, FRD, LN, LOG, RAD, SIN, SQR, TAN, TOD, TRN
3	**
4	- (부정), NOT
5	*, /, MOD
6	- (빼기), +
7	논리곱
8	XOR
9	또는

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정 명령어는 활성화되면 식을 평가하고 Dest 에 결과를 저장합니다.
사후 스캔	N/A

예제

래더 다이어그램

예 1

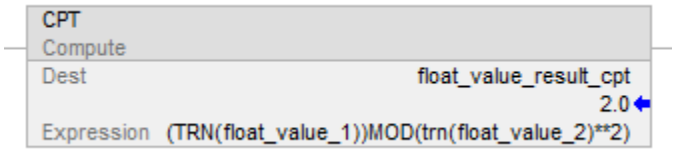
CPT 명령어는 활성화되면 value_1 에 5 를 곱하고 결과를 value_2 를 7 로 나눈 결과로 나눈 다음 result_1 에 최종 결과를 저장합니다.

result_1	25.0		Float	REAL
⊞ value_1	10		Decimal	DINT
⊞ value_2	14		Decimal	DINT

예 2

CPT 명령어는 활성화되면 float_value_1 과 float_value_2 를 2 제곱까지 자르고 잘라버린 float_value_1 을 그 결과로 나눈 다음 float_value_result_cpt 나눈셈 후에 나머지를 저장합니다.

래더 다이어그램



float_value_result_cpt	2.0		Float	REAL
float_value_1	10.5		Float	REAL
float_value_2	2.5		Float	REAL

추가 참조

계산 명령어 페이지의 411

유효한 연산자 페이지의 407

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

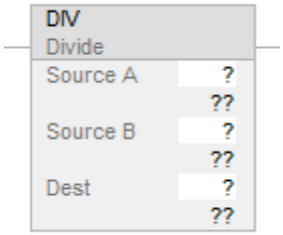
나누기(DIV)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, DIV 명령어와 '/' 연산자는 소스 A 를 소스 B 로 나눕니다.

사용 가능한 언어

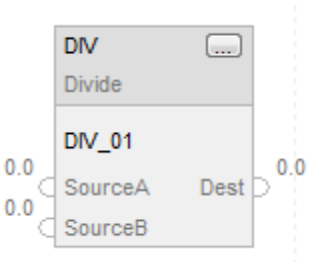
래더 다이어그램



함수 블록 다이어그램

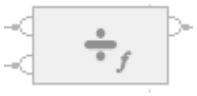
평션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.



ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 계산하려면 식에서 '/' 연산자를 사용하십시오. ST(스트럭처드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	형식	설명
SourceA	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	피제수 값.
SourceB	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	제수 값.
Dest	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	태그	명령어의 결과를 저장할 태그.

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
DIV	FBD_MATH	태그	DIV 구조

FBD_MATH 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
Source A	REAL	피제수의 값.
Source B	REAL	제수의 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	REAL	명령어의 결과.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러 데이터 유형	설명
SourceA(상위)	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	피제수의 값.

SourceB(하위)	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	제수 값.

출력 피연산자(오른쪽 핀)	**CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러 데이터 유형**	**설명**
Dest	DINT UDINT LINT ULINT REAL LREAL	펑션 결과

*FBD 펑션*을 참조하십시오.

연산 상태 플래그에 영향

컨트롤러	**연산 상태 플래그에 영향**
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

마이너 폴트 발생 조건:	**폴트 유형**	**폴트 코드**
Source_B = 0	4	4

배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 Dest = Source A / Source B [1,2]
사후 스캔	N/A

함수 블록 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	Dest = SourceA / SourceB [1,2] 오버플로가 발생하는 경우 EnableOut 을 거짓으로 해제 else EnableOut 을 참으로 설정
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

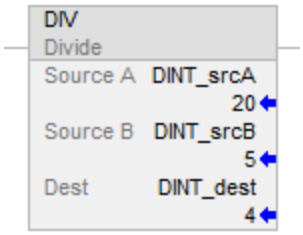
조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	Dest = SourceA / SourceB [1,2]
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

[1] Source B 가 0 이면 결과는 Source A 이고 마이너 폴트가 생성됩니다.

[2] 정수 대상 및 소스 피연산자의 결과가 잘립니다.

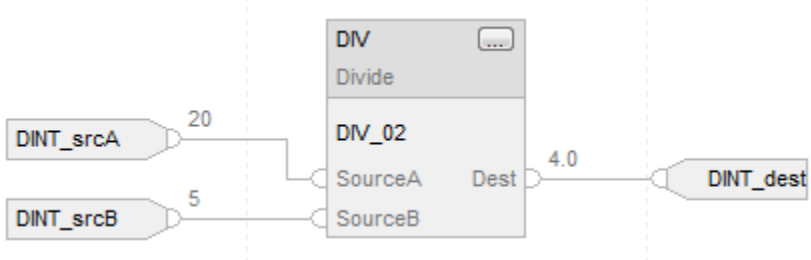
예제

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 평션

ST(스트럭쳐드 텍스트)

DINT_dst := DINT_srcA / DINT_srcB;

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

즉시 값 페이지의 967

FBD 펑션 페이지의 472

모듈로(MOD)

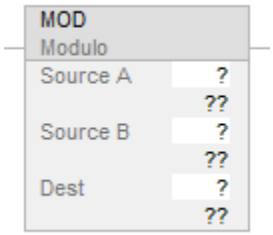
이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, MOD 명령어와 연산자는 소스 A 를 소스 B 로 나눈 후에 나머지를 Dest 에 표시합니다. 이는 다음 알고리즘을 사용하여 수행됩니다.

Dest = Source A – (truncate (Source A / Source B) * Source B)

사용 가능한 언어

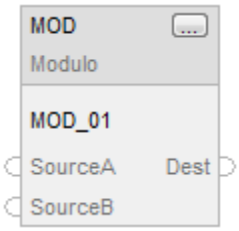
래더 다이어그램



함수 블록 다이어그램

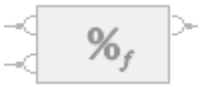
펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.



ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 계산하려면 식에서 MOD를 연산자로 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

다음은 래더 다이어그램에 대한 피연산자입니다.

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
Source A	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	피제수의 값.
Source B	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	제수의 값.
Dest	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	태그	명령어의 결과를 저장하는 태그.

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
MOD	FBD_MATH	태그	MOD 구조

FBD_MATH 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	REAL	피제수의 값.
SourceB	REAL	제수의 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	REAL	명령어의 결과.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	설명
SourceA(상위)	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	피제수의 값.

SourceB(하위)	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	제수 값.

출력 **피연산자(오른쪽 핀)**	**데이터 유형** **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	**설명**
Dest	DINT UDINT LINT ULINT REAL LREAL	펑션 결과.

*FBD 펑션*을 참조하십시오.

연산 상태 플래그에 영향

컨트롤러	**연산 상태 플래그에 영향**
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

마이너 폴트 발생 조건:	폴트 유형	폴트 코드
소스 B = 0	4	4

배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 설명 섹션에서 설명한 바와 같이 대상이 나머지 명령어로 설정됩니다.
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	설명 섹션에서 설명한 바와 같이 대상이 나머지 명령어로 설정됩니다. 오버플로가 발생할 경우 EnableOut 을 거짓으로 해제 else EnableOut 을 참으로 설정
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

FBD 펑션

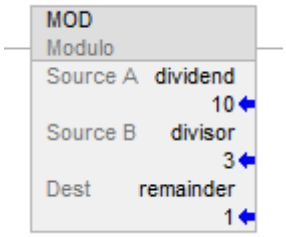
팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	설명 섹션에서 설명한 바와 같이 대상이 나머지 명령어로 설정됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

팁: Source B 가 0 이면 결과가 0 이고 마이너 폴트가 발생합니다.

예제

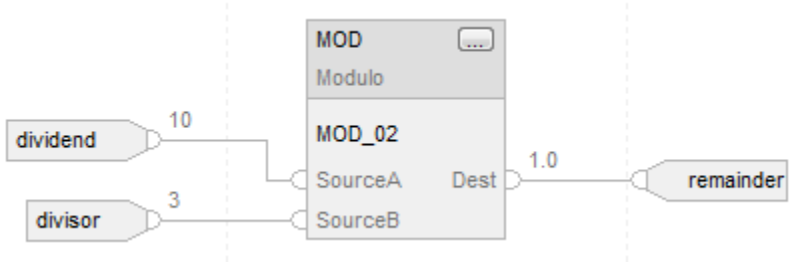
래더 다이어그램



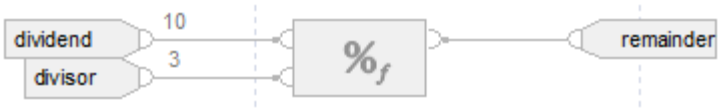
피제수를 제수로 나누고 나머지를 나머지에 놓습니다. 이 예에서는 10 을 3 으로 세 번 나눠 나머지가 1 이 됩니다.

함수 블록 다이어그램

FBD 블록



FBD 평션



ST(스트럭쳐드 텍스트)

remainder := dividend MOD divisor;

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

즉시 값 페이지의 967

FBD 펑션 페이지의 472

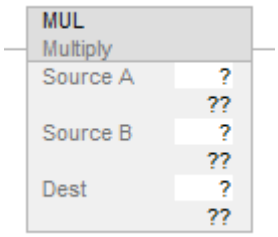
곱하기(MUL)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, MUL 명령어와 연산자 '*'는 소스 A 와 소스 B 를 곱합니다.

사용 가능한 언어

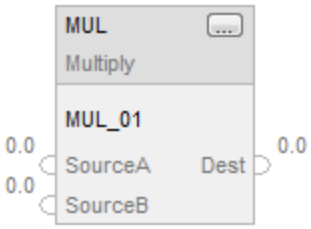
래더 다이어그램



함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 연산자 '*'를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형 **CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러**	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러**	형식	설명
Source A	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	피승수의 값.
Source B	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	승수의 값.

Dest	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	태그	명령어의 결과를 저장할 태그.

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
MUL	FBD_MATH	태그	MUL 구조

FBD_MATH 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	REAL	피승수의 값.
SourceB	REAL	승수의 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	REAL	명령어의 결과.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러 데이터 유형	설명
SourceA(상위)	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	피승수의 값.
SourceB(하위)	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	승수의 값.

출력 피연산자(오른쪽 핀)	CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러 데이터 유형	설명
Dest	DINT UDINT LINT ULINT REAL LREAL	펑션 결과.

*FBD 평션*을 참조하십시오.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 Dest = Source A x Source B
사후 스캔	N/A

함수 블록 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	Dest = SourceA x SourceB 오버플로가 발생하는 경우 EnableOut 을 거짓으로 해제 else EnableOut 을 참으로 설정
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

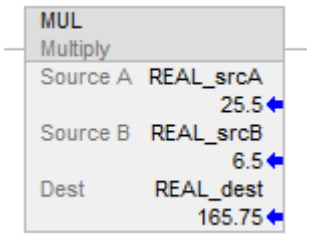
FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	Dest = Source A x Source B
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

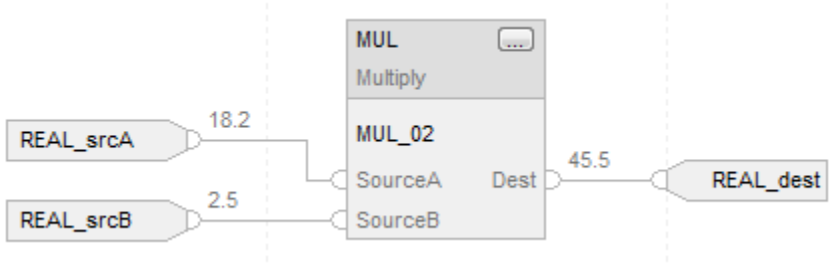
예제

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 펑션

ST(스트럭쳐드 텍스트)

```
REAL_dest := REAL_srcA * REAL_srcB;
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

즉시 값 페이지의 967

FBD 펑션 페이지의 472

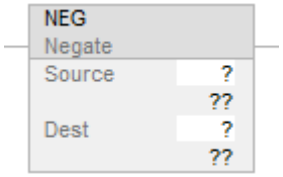
부정(NEG)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, NEG 명령어와 연산자는 0 에서 소스 값을 뺍니다.

사용 가능한 언어

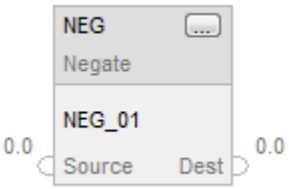
래더 다이어그램



함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 계산하려면 식에 '-' 연산자를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	형식	설명
소스	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	부정할 값.

Dest	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	태그	명령어의 결과를 저장하는 태그.

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
NEG	FBD_MATH_ADVANCED	태그	NEG 구조

FBD_MATH_ADVANCED 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
소스	REAL	부정할 값

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	REAL	명령어의 결과

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형	설명
소스	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	부정할 값

출력 피연산자(오른쪽 핀)	데이터 유형	설명
Dest	DINT UDINT LINT ULINT REAL LREAL	펑션 결과.

*FBD 펑션*을 참조하십시오.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. Dest = 0 - Source.
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 을 EnableIn 으로 설정합니다.
EnableIn 이 참임	Dest = 0 - Source. 오버플로가 발생하는 경우 EnableOut 을 거짓으로 해제 else EnableOut 을 참으로 설정
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

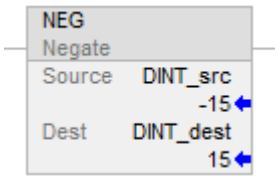
FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	Dest = 0 - Source.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

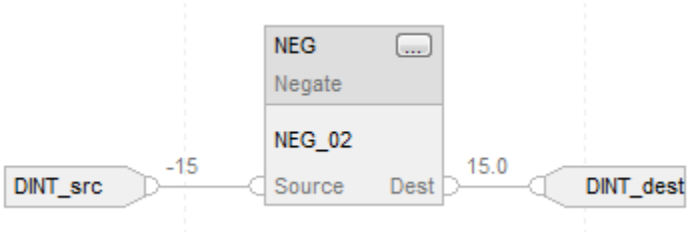
예제

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 평션

ST(스트럭처드 텍스트)

```
DINT_dest := -DINT_src;
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

FBD 펑션 페이지의 472

즉시 값 페이지의 967

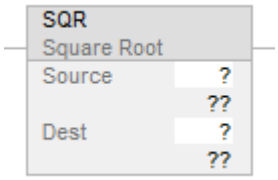
제곱근(SQR/SQRT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

SQR 명령어와 연산자는 소스의 제곱근을 계산하여 그 결과를 Dest 에 표시합니다.

사용 가능한 언어

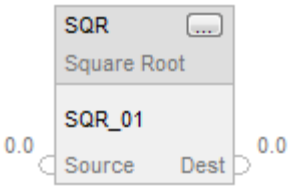
래더 다이어그램



함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 계산하려면 식에서 SQRT 를 연산자로 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 ST(스트럭쳐드 텍스트) 구문을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
소스	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	이 값의 제곱근을 계산합니다.
Dest	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	태그	명령어의 결과를 저장할 태그.

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
SQR	FBD_MATH_ADVANCED	태그	SQR 구조

FBD_MATH_ADVANCED 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
소스	REAL	이 값의 제곱근을 확인합니다.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	REAL	명령어의 결과.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	설명
SourceA	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	이 값의 제곱근을 계산합니다.

출력 피연산자(오른쪽 핀)	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	설명
Dest	DINT UDINT LINT ULINT REAL LREAL	펑션 결과.

FBD 펑션을 참조하십시오.

설명

Dest 가 LREAL/REAL 이 아닌 경우, 명령어는 결과의 분수 부분을 다음과 같이 처리합니다.

Source 는 다음인 경우	(CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러) 결과의 분수 부분	예			(CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러) 결과의 분수 부분	예		
모든 기본 정수 태그/값	자르기	소스	DINT	3	반올림	소스	DINT	3
		Dest	DINT	1		Dest	DINT	2
모든 부동 소수점 태그/값	반올림	소스	REAL	3.0	반올림	소스	REAL	3.0
		Dest	DINT	2		Dest	DINT	2

Source 가 음수인 경우, 명령어는 제곱근을 계산하기 전에 Source 의 절대값을 취합니다.

CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러의 경우, Source 가 정수 데이터 유형이고 Dest 가 정수 데이터 유형이면 명령어가 결과의 분수 부분을 잘라냅니다. 예를 들어, 정수의 Source 값이 3 이면 결과가 1.732 이고 Dest 값은 1 이 됩니다.

Source 가 REAL 데이터 Dest 이고 대상이 정수 유형인 경우에는 명령어가 결과를 반올림 처리합니다. 예를 들어, 실수 Source 값이 3.0 이면 결과가 1.732 이고 Dest 값은 2 가 됩니다.

SQR 은 래더 다이어그램 식에서 연산자로 사용되고, SQRT 는 ST(스트럭처드 텍스트) 구문에서 연산자로 사용됩니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. Dest = Source 의 제곱근.
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 을 EnableIn 으로 설정합니다.
EnableIn 이 참임	Dest. = Source 의 제곱근 오버플로가 발생하는 경우 EnableOut 을 거짓으로 해제 else EnableOut 을 참으로 설정
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

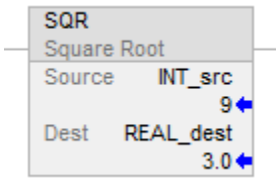
FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	Dest = 소스의 제곱근
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

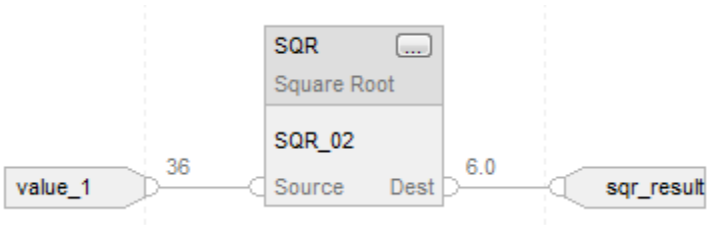
예제

래더 다이어그램



함수 블록 다이어그램

FBD 블록



FBD 펑션

ST(스트럭쳐드 텍스트)

REAL_dest := SQRT(INT_src);

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

데이터 변환 페이지의 967

연산 상태 플래그 페이지의 963

FBD 펑션 페이지의 472

즉시 값 페이지의 967

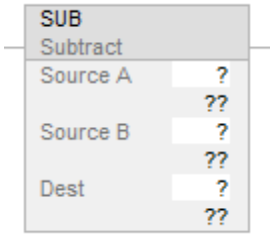
빼기(SUB)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

활성화된 경우, SUB 명령어와 '-' 연산자는 소스 A 에서 소스 B 를 뺍니다.

사용 가능한 언어

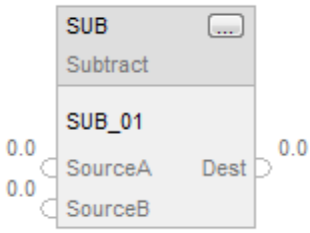
래더 다이어그램



함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 연산자 '-'를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	형식	설명
Source A	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	소스 B 를 뺄 값.
Source B	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	즉시 태그	소스 A 에서 뺄 값.
Dest	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	태그	명령어의 결과를 저장하는 태그.

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
SUB	FBD_MATH	태그	SUB 구조

FBD_MATH 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	REAL	SourceB 를 뺄 값.
SourceB	REAL	SourceA 에서 뺄 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	REAL	명령어의 결과.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러 데이터 유형	설명
SourceA(상위)	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	SourceB 를 뺄 값.

SourceB(하위)	SINT USINT INT UINT DINT UDINT LINT ULINT REAL LREAL	SourceA 에서 뺄 값.

출력 피연산자 (오른쪽 핀)	**CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러 데이터 유형**	**설명**
Dest	DINT UDINT LINT ULINT REAL LREAL	펑션 결과.

*FBD 펑션*을 참조하십시오.

연산 상태 플래그에 영향

컨트롤러	**연산 상태 플래그에 영향**
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 Dest = Source A - Source B
사후 스캔	N/A

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	Dest = SourceA - SourceB 오버플로가 발생하는 경우 EnableOut 을 거짓으로 해제 else EnableOut 을 참으로 설정
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

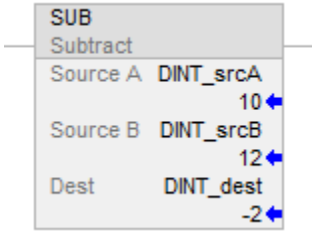
FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	Dest = SourceA - SourceB
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

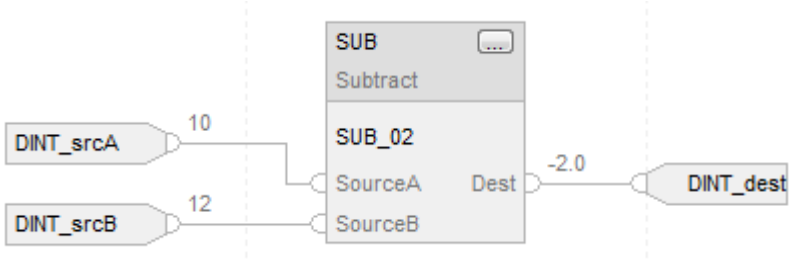
예제

래더 다이어그램

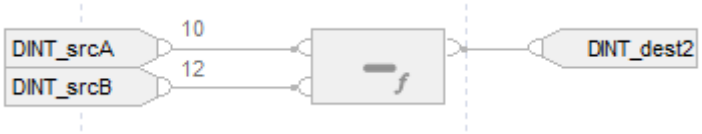


함수 블록 다이어그램

FBD 블록



FBD 평션



ST(스트럭처드 텍스트)

```
DINT_dest := DINT_srcA - DINT_srcB;
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

FBD 펑션 페이지의 472

즉시 값 페이지의 967

FBD 펑션

이 정보는 Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580 및 GuardLogix 5580 컨트롤러에 적용됩니다..

FBD 펑션은 IEC 61131-3 Edition 3 에 따라 구현됩니다. 산술 및 숫자 펑션은 펑션 블록 다이어그램 언어로 제공됩니다. 래더 다이어그램과 ST(스트럭쳐드 텍스트) 언어에는 산술 및 숫자가 연산자와 펑션으로 포함되어 있습니다.

FBD 펑션에는 한 개 이상의 입력과 한 개의 출력이 있습니다. FBD 펑션은 효율성을 제공할 수 있도록 구현되며 설치 공간이 작고 FBD 펑션 블록보다 더 적은 리소스를 사용하여 작동할 수 있습니다.

FBD 펑션

- 모든 입력과 출력이 필요합니다. 모든 입력이 지원되는 데이터 유형을 사용해야 합니다.
- 지원 태그나 미리 정의된 데이터 유형이 없습니다. 연결된 입력 값이 미리 정의된 데이터 유형으로 변환되지 않습니다.
- EnableIn 비트가 없으며 항상 실행됩니다.

예: 평션 추가

추가 참조

[펑션 오버로딩](#) 페이지의 473

[데이터 변환](#) 페이지의 967

펑션 오버로딩

이 정보는 Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580 및 GuardLogix 5580 컨트롤러에 적용됩니다.

펑션 오버로딩은 이름은 같지만 서명이 다른 2 개 이상의 펑션(인수 또는 반환 유형 등)을 정의합니다. 오버로딩을 지원하는 FBD 펑션은 다양한 입력 데이터 유형을 사용합니다. 출력 데이터 유형은 입력 데이터 유형에 따라 다릅니다.

FBD 펑션은 다음과 같은 규칙을 따릅니다.

- 입력 유형 승격
  - 입력 유형 승격
    - 가장 높은 우선순위에서 가장 낮은 우선순위 순으로 지정된 데이터 유형:
      LREAL, REAL, ULINT, LINT, UDINT, DINT, UINT, INT, USINT, SINT
    - 모든 입력이 실행 전에 순위가 가장 높은 입력 데이터 유형으로 승격됨
    - 모든 입력에 DINT 또는 그보다 낮은 순위 값이 있는 경우 모든 입력이 실행 전에 DINT 유형으로 승격됨
  - 출력 유형은 입력 유형에 따라 다름
    펑션의 출력 유형은 승격된 입력 유형임

예를 들어, 펑션 추가의 경우

- SINT + UINT 입력이 DINT + DINT 입력으로 승격됩니다. 출력이 DINT 인 경우
- USINT + LINT 입력이 LInt + LINT 입력으로 승격됩니다. 출력이 LINT 인 경우

- UNIT + LREAL 입력이 LREAL + LREAL 입력으로 승격됩니다. 출력이 LREAL 인 경우

추가 참조

FBD 펑션 페이지의 472

데이터 변환 페이지의 967

이동/로직 명령어

이동/로직 명령어

이동 명령어를 사용하여 비트를 수정하고 이동할 수 있습니다.

사용 가능한 명령어

래더 다이어그램

MOV	MVM	논리곱	또는	XOR	NOT	SWPB	CLR	BTD

함수 블록 다이어그램

FBD 블록

MVMT	논리곱	또는	XOR	NOT	BTDT	BAND	BXOR

BNOT	BOR

FBD 펑션

BNOT	BOR	BAND	BXOR

ST(스트럭처드 텍스트)

MVMT	SWPB	BTDT

실행할 작업:	사용할 명령어:
값을 복사하거나 문자열을 이동합니다.	MOV
정수의 특정 부분을 복사합니다.	MVM

펑션 블록에서 정수의 특정 부분을 복사합니다.	MVMT
정수 내에서 또는 정수 사이에서 비트를 이동합니다.	BTD
펑션 블록에서 정수 내 또는 정수 사이에서 비트를 이동합니다.	BTDT
값을 지웁니다.	CLR
INT, DINT 또는 REAL 태그의 바이트를 다시 정렬합니다.	SWPB

로직 명령어가 비트에서 로직 연산을 수행합니다.

실행할 작업:	사용할 명령어:
비트별 논리곱 연산을 수행합니다.	논리곱
비트별 논리합 연산을 수행합니다.	또는
비트별 배타적 논리합 연산을 수행합니다.	XOR
비트별 논리부정 연산을 수행합니다.	NOT

데이터 유형을 혼합 사용할 수 있지만 정확도가 떨어지고 반올림 에러가 발생할 수 있으며 명령어를 실행하는 데 시간이 더 오래 걸립니다. 결과가 잘렸는지 확인하려면 S:V 비트를 확인하십시오.

볼드체 데이터 유형은 최적의 데이터 유형을 나타냅니다. 명령어의 모든 피연산자가 동일한 최적의 데이터 유형(일반적으로 DINT 또는 REAL)을 사용하는 경우 명령어는 가장 적은 메모리를 사용하면서 보다 빠르게 실행됩니다.

이동/로직 명령어는 렁-입력-조건이 참인 경우 명령어가 스캔될 때마다 한 번 실행됩니다. 식을 한 번만 계산하려면 원샷 명령어를 사용하여 이동/로직 명령어를 트리거하십시오.

추가 참조

연산 변환 명령어 페이지의 837

입력/출력 명령어 페이지의 165

반복/중단 명령어 페이지의 727

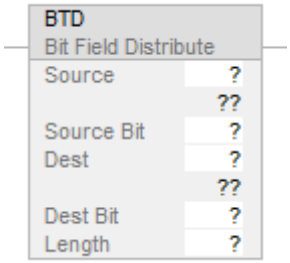
비트 필드 분산(BTD)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

BTD 명령어는 지정된 비트를 Source 에서 복사해 해당 위치로 이동한 다음 Dest 에 기록합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT	즉시 태그	이동할 비트를 포함한 태그
Source bit	DINT	즉시(0 ~ 31)	이동을 시작할 비트 번호(가장 낮은 비트 번호) 유효한 Source 데이터 유형 범위 내에 있어야 함

Destination	SINT INT DINT	태그	비트를 이동할 태그
Destination bit	DINT	즉시(0 ~ 31)	비트 번호(이로 데이터를 이동해야 함)가 유효한 Destination 데이터 유형 범위 내에 있어야 합니다.
Length	DINT	즉시(1-32)	이동할 비트 수

설명

활성화된 경우 BTD 명령어는 비트 그룹을 Source 에서 Destination 으로 복사합니다. 비트 그룹은 Source bit(Source 의 가장 낮은 비트 번호)와 Length(복사할 비트 수)로 식별됩니다. Destination bit 는 Destination 에서 시작할 가장 낮은 비트 번호를 식별합니다. 소스는 변경되지 않습니다.

비트 필드의 길이가 Destination 을 넘어 확장될 경우 이 명령어는 추가 비트를 저장하지 않습니다. 추가 비트가 다음 워드로 래핑되지 않습니다.

SINT 또는 INT 태그는 영 채우기를 통해 DINT 값으로 변환됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

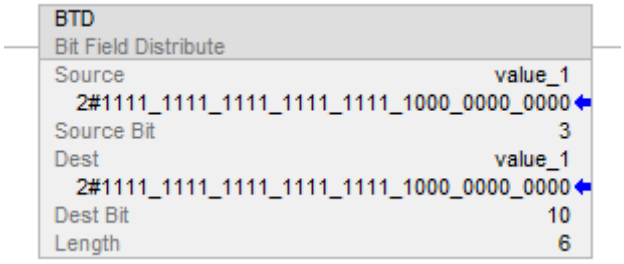
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임.	N/A
렁-입력-조건이 참임.	명령어가 Source 비트를 복사하여 Destination 으로 이동합니다.
사후 스캔	N/A

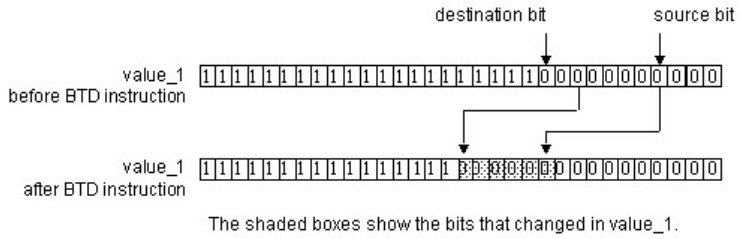
예

예 1

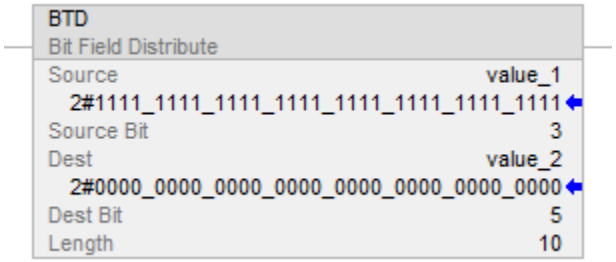
래더 다이어그램



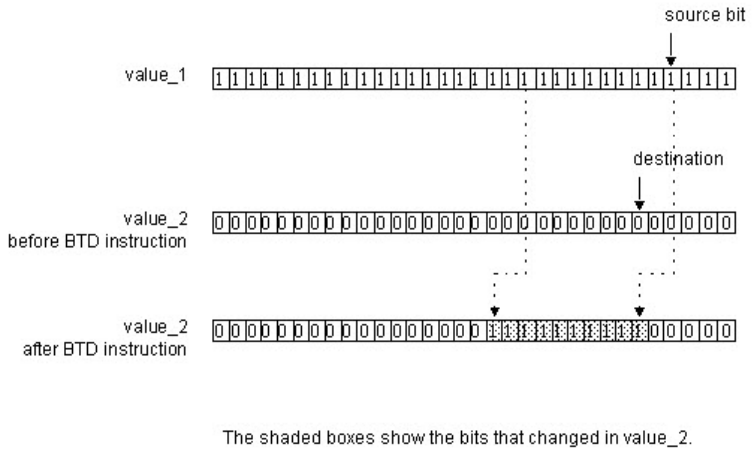
활성화된 경우 BTD 명령어는 value_1 내에서 비트를 이동합니다.



예 2



활성화된 경우 BTD 명령어는 10 비트를 value_1 에서 value_2 로 이동합니다.



The shaded boxes show the bits that changed in value_2.

추가 참조

이동 명령어 페이지의 475

지우기(CLR) 페이지의 525

공통 속성 페이지의 963

데이터 변환 페이지의 967

마스킹된 이동(MVM) 페이지의 527

비트 필드 분산(대상 포함)(BTDT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

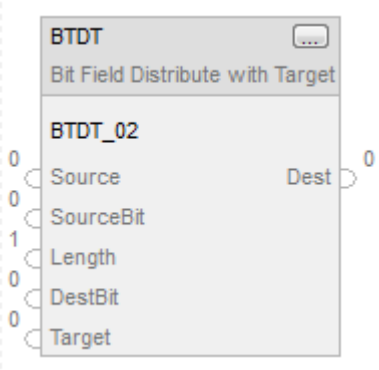
BTDT 명령어는 먼저 Target 을 Destination 에 복사합니다. 그런 다음 이 명령어는 지정된 비트를 Source 에서 복사해 해당 위치로 이동한 다음 Destination 에 기록합니다. Target 과 Source 는 그대로 유지됩니다.

사용 가능한 언어

래더 다이어그램

이 명령어는 래더 다이어그램에서 사용할 수 없습니다.

평션 블록



ST(스트럭처드 텍스트)

BTDT(BTDT_tag);

피연산자

평션 블록

피연산자	유형	형식	설명
BTDT tag	FBD_BIT_FIELD_DISTRIBUTE	구조	BTDT 구조

ST(스트럭처드 텍스트)

입력 파라미터	데이터 유형	설명
활성화 입력	BOOL	선택하지 않으면 명령이 실행되지 않고 출력이 업데이트되지 않습니다. 설정되면 명령어가 실행됩니다. 기본값은 설정 상태입니다.
소스	DINT	Destination 으로 이동할 비트가 포함된 입력값. 유효값 = 모든 정수
SourceBit	DINT	Source 의 비트 위치(이동을 시작할 가장 낮은 비트 번호). 유효값 = 0 ~ 31
길이(Length)	DINT	이동할 비트 수. 유효값 = 1 ~ 32

DestBit	DINT	Dest 의 비트 위치(비트 복사를 시작할 가장 낮은 비트 번호). 유효값 = 0 ~ 31
Target	DINT	Source 에서 비트를 이동하기 전에 Dest 로 이동할 입력값. 유효값 = 모든 정수

출력 파라미터	데이터 유형	설명
활성화 출력(EnableOut)	BOOL	명령어의 활성화 여부를 나타냅니다.
Dest	DINT	비트 이동 작업의 결과.

ST(스트럭쳐드 텍스트) 내에서 식의 구문에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

설명

참인 경우 BTDT 명령어는 먼저 Target 을 Destination 에 복사하고 비트 그룹을 Source 에서 Destination 으로 복사합니다. 비트 그룹은 Source bit(그룹의 가장 낮은 비트 번호)와 Length(복사할 비트 수)로 식별됩니다. Destination bit 는 Destination 에서 시작할 가장 낮은 비트 번호 비트를 식별합니다. Source 와 Target 은 그대로 유지됩니다.

비트 필드의 길이가 Destination 을 넘어 확장될 경우 이 명령어는 추가 비트를 저장하지 않습니다. 추가 비트가 다음 워드로 래핑되지 않습니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	예
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	아니요

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

펑션 블록

조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 명령어가 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

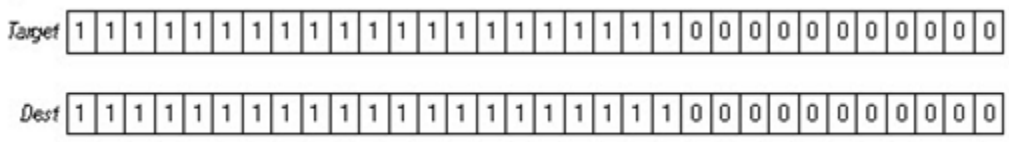
ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	펑션 블록 표의 사전 스캔을 참조하십시오.
정상 실행	펑션 블록 표에서 Tag.EnableIn 이 참임을 참조하십시오.
사후 스캔	펑션 블록 표에서 사후 스캔을 참조하십시오.

예

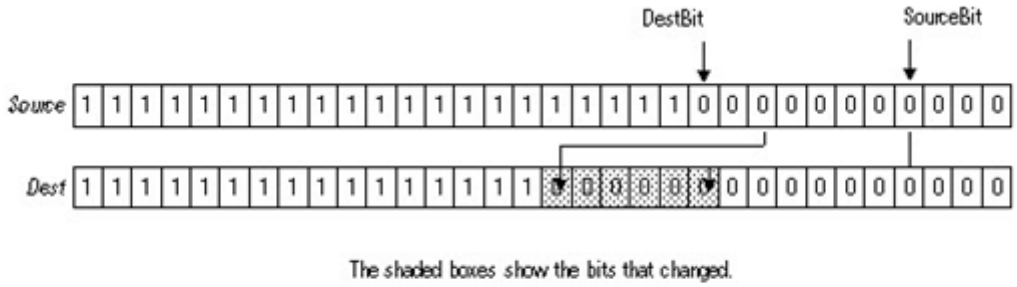
1 단계

컨트롤러가 Target 을 Dest 에 복사합니다.



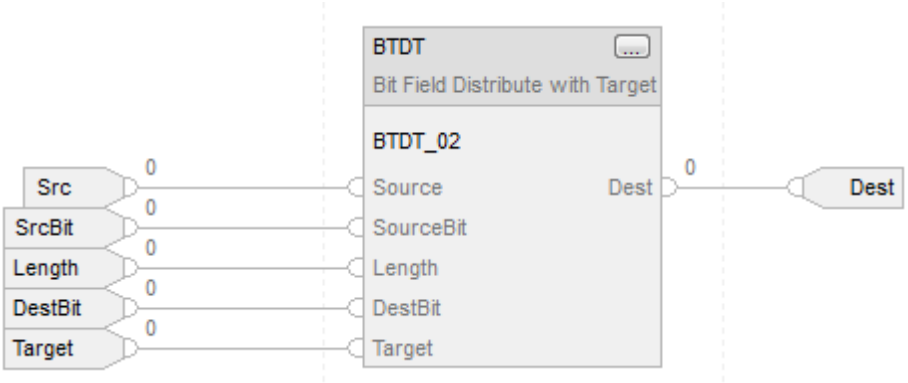
2 단계

SourceBit 와 Length 가 Dest 에 복사할 Source 의 비트를 지정합니다. DestBit 에서 시작하여 Source 와 Target 은 그대로 유지됩니다.



The shaded boxes show the bits that changed.

펑션 블록



ST(스트럭처드 텍스트)

```
BTDT_01.Source := sourceSTX;

BTDT_01.SourceBit := source_bitSTX;

BTDT_01.Length := LengthSTX;

BTDT_01.DestBit := dest_bitSTX;

BTDT_01.Target := TargetSTX;

BTDT(BTDT_01);

distributed_value := BTDT_01.Dest;
```

추가 참조

공통 특성 페이지의 963

연산 상태 플래그 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

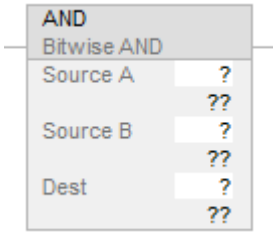
비트별 논리곱(AND)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

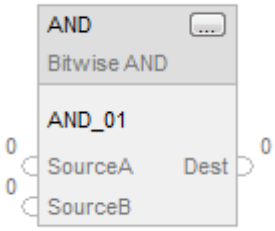
AND 명령어는 Source A 와 Source B 의 비트를 사용하여 비트별 논리곱 연산을 수행한 후에 그 결과를 Dest 에 표시합니다.

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 계산하려면 식 내에 연산자 AND(또는 '&')를 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Source A	SINT INT DINT REAL	즉시 태그	Source B 와 논리곱 연산 수행할 값. **팁:** 데이터 유형이 REAL 인 경우, 입력값이 DINT 로 변환되면서 오버플로가 발생할 수 있습니다.
Source B	SINT INT DINT REAL	즉시 태그	Source A 와 논리곱 연산 수행할 값. **팁:** 데이터 유형이 REAL 인 경우, 입력값이 DINT 로 변환되면서 오버플로가 발생할 수 있습니다.
Dest	SINT INT DINT REAL	태그	명령어의 결과를 저장하는 태그. **팁:** 데이터 유형이 REAL 인 경우, 결과인 DINT 값은 REAL 로 변환됩니다.

팁: AND 명령어는 DINT 에 대해 연산합니다. INT 또는 SINT 소스 피연산자는 상위 비트를 0 으로 채워 DINT 로 변환됩니다.

평션 블록

피연산자	데이터 유형	형식	설명
AND	FBD_LOGICAL	태그	AND 구조

FBD_LOGICAL 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	DINT	SourceB 와 논리곱 연산 수행할 값.
SourceB	DINT	SourceA 와 논리곱 연산 수행할 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화된 경우에 실패 없이 실행되었는지의 여부를 나타냅니다.
Dest	DINT	명령어의 결과.

설명

활성화된 경우 이 명령어는 비트별 논리곱 연산을 계산합니다.
Dest = A AND B

Source A 의 비트:	Source B 의 비트:	Dest 의 비트:
0	0	0
0	1	0
1	0	0
1	1	1

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

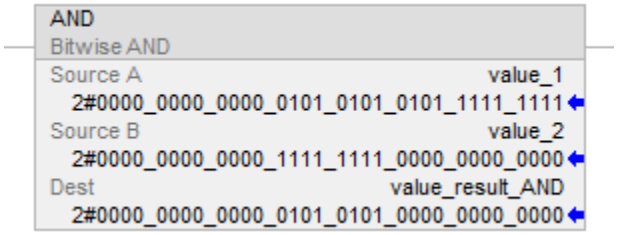
조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 Dest 는 설명 섹션에서 설명한 대로 설정됩니다.
사후 스캔	N/A

펑션 블록

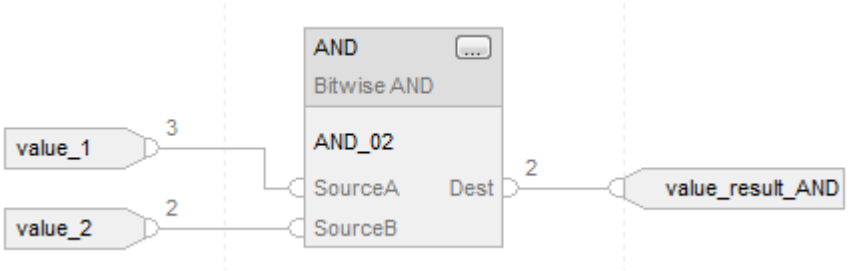
조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	EnableOut 을 EnableIn 으로 설정합니다. Dest 는 설명 섹션에서 설명한 대로 설정됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

예

래더 다이어그램



평션 블록



ST(스트럭쳐드 텍스트)

value_result_and := value_1 AND value_2;

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

이동 명령어 페이지의 475

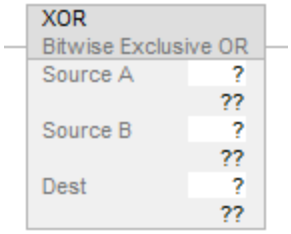
비트별 배타적 논리합(XOR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

XOR 명령어는 Source A 와 Source B 의 비트를 사용하여 비트별 배타적 논리합 연산을 수행하고 결과를 Dest 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 연산자로 XOR 을 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Source A	SINT INT DINT REAL	즉시 태그	Source B 와 배타적 논리합 연산 수행할 값. **팁:** 유형이 REAL 인 경우 입력값이 DINT 로 변환됩니다(오버플로를 유발할 수 있음).
Source B	SINT INT DINT REAL	즉시 태그	Source A 와 배타적 논리합 연산 수행할 값. **팁:** 유형이 REAL 인 경우 입력값이 DINT 로 변환됩니다(오버플로를 유발할 수 있음).
Dest	SINT INT DINT REAL	태그	명령어의 결과를 저장하는 태그. **팁:** 유형이 REAL 인 경우 결과 DINT 값이 REAL 로 변환됩니다.

팁: XOR 명령어는 DINT 에 대해 연산합니다. INT 또는 SINT 소스 피연산자는 상위 비트를 0 으로 채워 DINT 로 변환됩니다.

평션 블록

피연산자	데이터 유형	형식	설명
XOR	FBD_LOGICAL	태그	XOR 구조

FBD_LOGICAL 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.

SourceA	DINT	SourceB 와 배타적 논리합 연산 수행할 값.
SourceB	DINT	SourceA 와 배타적 논리합 연산 수행할 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	DINT	명령어의 결과.

설명

활성화된 경우 이 명령어는 비트별 배타적 논리합 연산을 평가합니다.

Dest = Source A XOR Source B

Source A 의 비트:	Source B 의 비트:	Dest 의 비트:
0	0	0
0	1	1
1	0	1
1	1	0

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

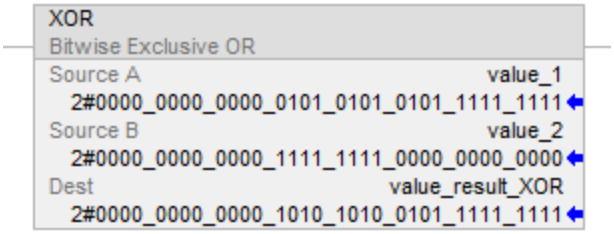
조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 Dest 는 설명 섹션에서 설명한 대로 설정됩니다.
사후 스캔	N/A

펑션 블록

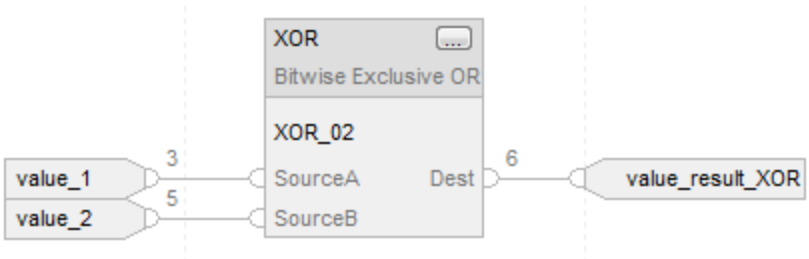
조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	EnableOut 에서 EnableIn 으로 설정 Dest 는 설명 섹션에서 설명한 대로 설정됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

예

래더 다이어그램



평션 블록



ST(스트럭쳐드 텍스트)

value_result_XOR := value_1 XOR value_2;

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

이동 명령어 페이지의 475

데이터 변환 페이지의 967

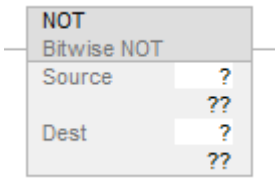
비트별 논리부정(NOT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

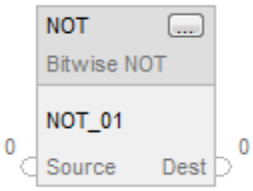
NOT 명령어는 Source 의 비트별 반전 작업을 수행하여 그 결과를 Dest 에 표시합니다.

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 계산하려면 식에서 NOT 를 연산자로 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형	형식	설명
소스	SINT INT DINT REAL	immediate 태그	논리부정 연산 수행할 값 **팁:** 유형이 REAL 인 경우 입력값이 DINT 로 변환됩니다(오버플로를 유발할 수 있음).
Dest	SINT INT DINT REAL	태그	명령어의 결과를 저장하는 태그. **팁:** 유형이 REAL 인 경우 결과 DINT 값이 REAL 로 변환됩니다.

팁: NOT 명령어는 DINT 에 대해 연산합니다. INT 또는 SINT 소스 피연산자는 상위 비트를 0 으로 채워 DINT 로 변환됩니다.

펑션 블록

피연산자	데이터 유형	형식	설명
NOT	FBD_CONVERT	태그	NOT 구조

FBD_CONVERT 구조

입력 구성원	데이터 유형	설명
활성화 입력	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
소스	DINT	논리부정 연산 수행할 값

출력 구성원	데이터 유형	설명
활성화 출력(Enable Out)	BOOL	명령어가 활성화되었을 때 폴트 없이 실행되었는지를 나타냅니다.
Dest	DINT	명령어의 결과

설명

활성화된 경우 이 명령어는 비트별 NOT 연산을 평가합니다.

Dest = NOT Source

소스의 비트:	Dest 의 비트:
0	1
1	0

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

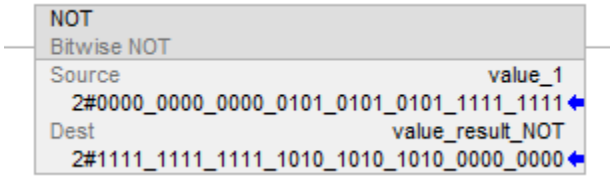
조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 Dest 는 설명 섹션에서 설명한 대로 설정됩니다.
사후 스캔	N/A

평션 블록

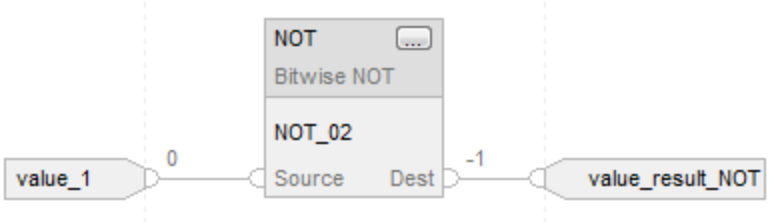
조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	EnableOut 을 EnableIn 으로 설정합니다. Dest 는 설명 섹션에서 설명한 대로 설정됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

예제

래더 다이어그램



평션 블록



ST(스트럭쳐드 텍스트)

value_result_NOT := NOT value_1;

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

이동 명령어 페이지의 475

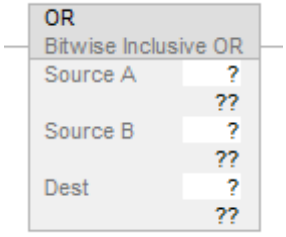
비트별 논리합(OR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

OR 명령어는 Source A 와 Source B 의 비트를 사용하여 비트별 OR 연산을 수행하고 결과를 Dest 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 연산자로 OR 을 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source A	SINT INT DINT REAL	즉시 태그	Source B 와 논리합 연산 수행할 값. **팁:** 유형이 REAL 인 경우 입력값이 DINT 로 변환됩니다(오버플로를 유발할 수 있음).

피연산자	유형	형식	설명
Source B	SINT INT DINT REAL	즉시 태그	Source A 와 논리합 연산 수행할 값. **팁:** 유형이 REAL 인 경우 입력값이 DINT 로 변환됩니다(오버플로를 유발할 수 있음).
Dest	SINT INT DINT REAL	태그	명령어의 결과를 저장하는 태그. **팁:** 유형이 REAL 인 경우 결과 DINT 값이 REAL 로 변환됩니다.

팁: OR 명령어는 DINT 에 대해 연산합니다. INT 또는 SINT 소스 피연산자는 상위 비트를 0 으로 채워 DINT 로 변환됩니다.

평션 블록

피연산자	유형	형식	설명
OR	FBD_LOGICAL	태그	OR 구조

FBD_LOGICAL 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
SourceA	DINT	SourceB 와 논리합 연산 수행할 값.
SourceB	DINT	SourceA 와 논리합 연산 수행할 값.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화된 경우 성공적으로 실행되었는지를 나타냅니다.
Dest	DINT	명령어의 결과.

설명

활성화된 경우 이 명령어는 비트별 논리합 연산을 평가합니다.

Dest = Source A OR Source B

Source A 의 비트:	Source B 의 비트:	Dest 의 비트:
0	0	0
0	1	1
1	0	1
1	1	1

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

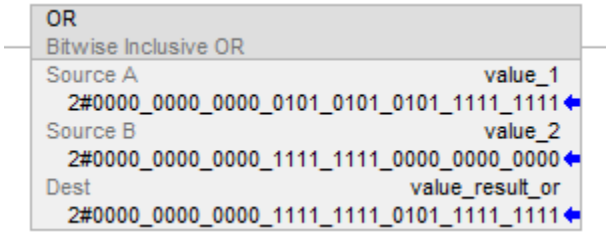
조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건에서 렁-입력-조건으로 설정
렁-입력-조건이 참임	렁-출력-조건에서 렁-입력-조건으로 설정 Dest 는 설명 섹션에서 설명한 대로 설정됩니다.
사후 스캔	N/A

평션 블록

조건/상태	취해진 조치
사전 스캔	N/A
EnableIn 이 거짓임	EnableOut 에서 EnableIn 으로 설정
EnableIn 이 참임	EnableOut 에서 EnableIn 으로 설정 Dest 는 설명 섹션에서 설명한 대로 설정됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

예

래더 다이어그램



평션 블록

ST(스트럭처드 텍스트)

```
value_result_or := value_1 OR value_2;
```

추가 참조

ST(스트럭처드 텍스트) 구문 페이지의 997

배열을 통한 인덱스 페이지의 978

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

이동 명령어 페이지의 475

부울 논리곱(BAND)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

BAND 명령어가 최대 8 개의 부울에 대해 논리곱 연산을 수행합니다. 비트별 논리곱을 수행하려면 *비트별 논리곱(AND)*를 참조하십시오.

사용 가능한 언어

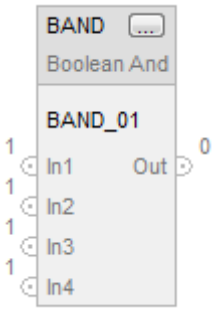
래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 2 개의 입력만 지원하며 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
BAND tag	FBD_BOOLEAN_AND	구조	BAND 구조

FBD_BOOLEAN_AND 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
In1	BOOL	첫 번째 부울 입력. 처음 다운로드 시 1 로 설정.
In2	BOOL	두 번째 부울 입력. 처음 다운로드 시 1 로 설정.
In3	BOOL	세 번째 부울 입력. 처음 다운로드 시 1 로 설정.
In4	BOOL	네 번째 부울 입력. 처음 다운로드 시 1 로 설정.
In5	BOOL	다섯 번째 부울 입력. 처음 다운로드 시 1 로 설정.

In6	BOOL	여섯 번째 부울 입력. 처음 다운로드 시 1 로 설정.
In7	BOOL	일곱 번째 부울 입력. 처음 다운로드 시 1 로 설정.
In8	BOOL	여덟 번째 부울 입력. 처음 다운로드 시 1 로 설정.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
Out	BOOL	명령어에 대한 출력.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	설명
In1	BOOL	첫 번째 부울 입력
In2	BOOL	두 번째 부울 입력

출력 피연산자(오른쪽 핀)	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	설명
Out	BOOL	명령어에 대한 출력.

FBD 펑션을 참조하십시오.

연산

FBD 블록

BAND 명령어가 최대 8 개의 부울 입력에 대해 논리곱 연산을 수행합니다. 입력을 사용하지 않을 경우 기본적으로 설정(1)으로 지정됩니다.

Out = In1 AND In2 AND In3 AND In4 AND In5 AND In6 AND In7 AND In8

중요: 편집 도중 BAND 명령어에서 입력 와이어를 제거할 경우 입력이 설정되었는지(1) 확인하십시오.

FBD 펑션

팁: FBD 펑션은 2 개의 입력만 지원하며 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

FBD 펑션이 2 개의 부울 입력에 대해 논리곱 연산을 수행합니다.

Out = In1 AND In2

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다.

실행

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 이 명령어는 작동 섹션에서 설명한 바와 같이 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

FBD 평션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	Out = In1 AND In2
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

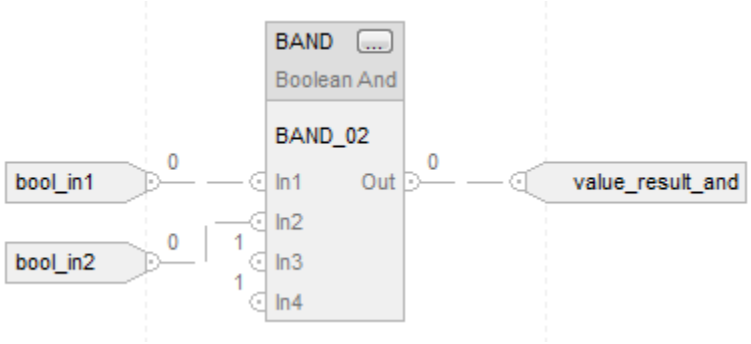
예

함수 블록 다이어그램

FBD 블록

이 예에서는 bool_in1 을 BAND_02.In1 에 복사하고 bool_in2 를 BAND_02.In2 에 복사하며 모든 BAND_02 입력에 논리곱 연산을 수행한 결과를 BAND_02.Out 에 저장한 다음 BAND_02.Out 을 value_result_and 에 복사합니다.

bool_in1 값:	bool_in2 값:	value_result_and:
0	0	0
0	1	0
1	0	0
1	1	1



FBD 평션

이 예는 bool_in1 과 bool_in2 에 대해 논리곱 연산을 수행하고 결과를 value_result_and 에 저장하는 방법을 보여줍니다.

추가 참조

비트별 논리곱(AND) 페이지의 485

FBD 평션 페이지의 472

부울 배타적 논리합(BXOR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

BXOR 명령어는 2 개의 부울 입력에 대해 배타적 논리합을 수행합니다.

사용 가능한 언어

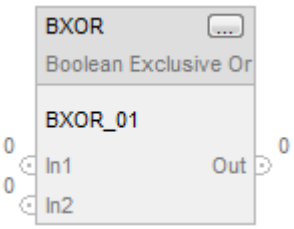
래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
BXOR tag	FBD_BOOLEAN_XOR	Structure	BXOR 구조

FBD_BOOLEAN_XOR 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
In1	BOOL	첫 번째 부울 입력. 기본값은 해제 상태입니다.
In2	BOOL	두 번째 부울 입력. 기본값은 해제 상태입니다.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
Out	BOOL	명령어에 대한 출력.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	설명
In1	BOOL	첫 번째 부울 입력.
In2	BOOL	두 번째 부울 입력.

출력 피연산자(오른쪽 핀)	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	설명
Out	BOOL	명령어에 대한 출력.

FBD 평션을 참조하십시오.

연산

BXOR 명령어는 2 개의 부울 입력에 대해 배타적 논리합을 수행합니다.

Out = In1 XOR In2

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다.

실행

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 이 명령어는 작동 섹션에서 설명한 바와 같이 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	Out = In1 XOR In2
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

예

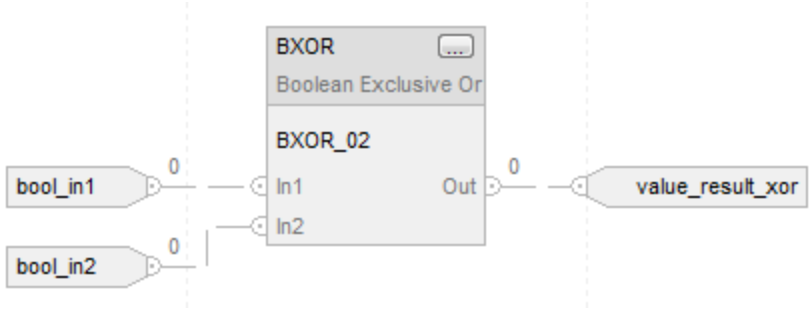
함수 블록 다이어그램

이 예에서는 bool_in1 을 BXOR_02.In1 에 복사하고 bool_in2 를 BXOR_02.In2 에 복사하며 BXOR_02.In1 및 BXOR_02.In2 에 대해 배타적 논리합 연산을 수행한 결과를 BXOR_02.Out 에 저장한 다음 BXOR_02.Out 을 value_result_xor 에 복사합니다.

bool_in1 값:	bool_in2 값:	value_result_xor 값:
0	0	0
0	1	1
1	0	1
1	1	0

FBD 블록

이 예는 bool_in1 과 bool_in2 에 대해 배타적 논리합을 수행하고 결과를 value_result_xor 에 저장하는 방법을 보여줍니다.



FBD 평션

추가 참조

비트별 배타적 논리합(XOR) 페이지의 489

FBD 평션 페이지의 472

부울 논리부정(BNOT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

BNOT 명령어는 부울 입력에 대한 보수 연산을 수행합니다. 비트별 논리부정을 수행하려면 *비트별 논리부정(NOT)*를 참조하십시오.

사용 가능한 언어

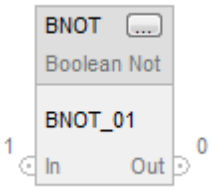
래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

함수 블록 다이어그램

평션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 평션

팁: FBD 평션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
BNOT tag	FBD_BOOLEAN_NOT	구조	BNOT 구조

FBD_BOOLEAN_NOT 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
In	BOOL	명령어에 대한 입력. 처음 다운로드 시 1 로 설정

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
Out	BOOL	명령어에 대한 출력.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 **CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러**	설명
In	BOOL	명령어에 대한 입력.

출력 피연산자(오른쪽 핀)	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	설명
Out	BOOL	명령어에 대한 출력.

FBD 평션을 참조하십시오.

연산

BNOT 명령어는 부울 입력에 대한 보수 연산을 수행합니다.

Out = NOT In

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다.

실행

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 이 명령어는 작동 섹션에서 설명한 바와 같이 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	이 명령어는 작동 섹션에서 설명한 바와 같이 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

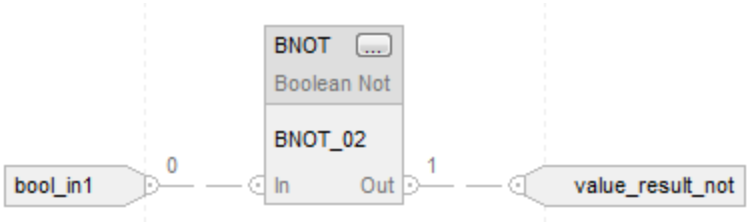
예

함수 블록 다이어그램

FBD 블록

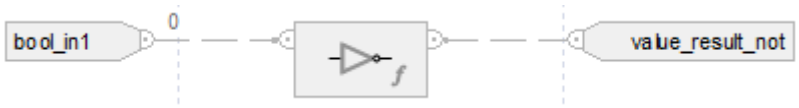
이 예에서는 bool_in1 을 BNOT_02.In 에 복사하고, BNOT_02.In 의 보수의 결과를 BNOT_02.Out 에 저장한 다음 BNOT_02.Out 을 value_result_not 에 복사합니다.

bool_in1 값:	value_result_not 값:
0	1
1	0



FBD 펑션

이 예에서는 bool_in1 의 보수의 결과를 value_result_not 에 저장합니다.



추가 참조

비트별 논리부정(NOT) 페이지의 494

FBD 펑션 페이지의 472

부울 논리합(BOR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

BOR 명령어가 최대 8 개의 부울 입력에 대해 논리합 연산을 수행합니다. 비트별 논리합을 수행하려면 *비트별 논리합(OR)*을 참조하십시오.

사용 가능한 언어

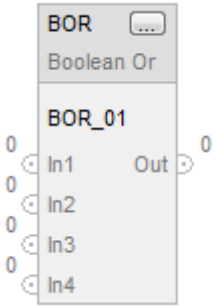
래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

함수 블록 다이어그램

펑션 블록 다이어그램은 다음 요소를 지원합니다.

FBD 블록



FBD 펑션

팁: FBD 펑션은 2 개의 입력만 지원하며 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

피연산자

함수 블록 다이어그램

FBD 블록

피연산자	데이터 유형	형식	설명
BOR tag	FBD_BOOLEAN_OR	구조	BOR 구조

FBD_BOOLEAN_OR 구조

입력 구성원	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제되면 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 처음 다운로드 시 0 으로 설정합니다.
In1	BOOL	첫 번째 부울 입력. 처음 다운로드 시 0 으로 설정합니다.
In2	BOOL	두 번째 부울 입력. 처음 다운로드 시 0 으로 설정합니다.
In3	BOOL	세 번째 부울 입력. 처음 다운로드 시 0 으로 설정합니다.

In4	BOOL	네 번째 부울 입력. 처음 다운로드 시 0 으로 설정합니다.
In5	BOOL	다섯 번째 부울 입력. 처음 다운로드 시 0 으로 설정합니다.
In6	BOOL	여섯 번째 부울 입력. 처음 다운로드 시 0 으로 설정합니다.
In7	BOOL	일곱 번째 부울 입력. 처음 다운로드 시 0 으로 설정합니다.
In8	BOOL	여덟 번째 부울 입력. 처음 다운로드 시 0 으로 설정합니다.

출력 구성원	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
Out	BOOL	명령어에 대한 출력.

FBD 펑션

팁: FBD 펑션은 2 개의 입력만 지원하며 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

입력 피연산자(왼쪽 핀)	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	설명
In1	BOOL	첫 번째 부울 입력.
In2	BOOL	두 번째 부울 입력.

출력 피연산자(오른쪽 핀)	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	설명
Out	BOOL	명령어에 대한 출력.

FBD 펑션을 참조하십시오.

연산

FBD 블록

BOR 명령어가 최대 8 개의 부울 입력에 대해 논리합 연산을 수행합니다. 입력을 사용하지 않을 경우 기본적으로 해제(0)로 지정됩니다.

Out = In1 OR In2 OR In3 OR In4 OR In5 OR In6 OR In7 OR In8

중요: 편집 도중 BOR 명령어에서 입력 와이어를 제거할 경우 입력이 해제되었는지(0) 확인하십시오.

FBD 펑션

팁: FBD 펑션은 2 개의 입력만 지원하며 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

FBD 펑션이 2 개의 부울 입력에 대해 논리합 연산을 수행합니다.

Out = In1 OR In2

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다.

실행

함수 블록 다이어그램

FBD 블록

조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 이 명령어는 작동 섹션에서 설명한 바와 같이 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

FBD 펑션

팁: FBD 펑션은 2 개의 입력만 지원하며 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용됩니다.

조건/상태	취해진 조치
사전 스캔	N/A
정상 스캔	Out = In1 OR In2
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	N/A

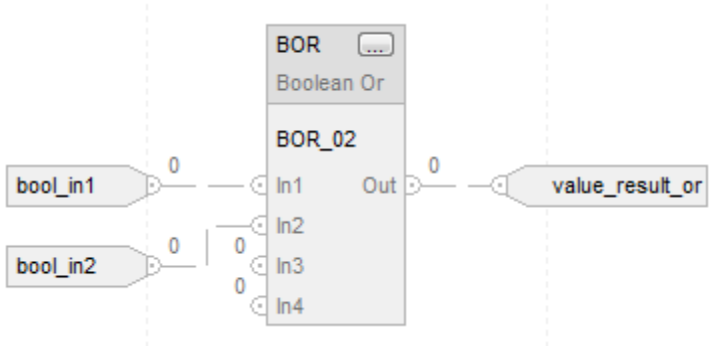
예

함수 블록 다이어그램

FBD 블록

이 예에서는 bool_in1 을 BOR_02.In1 에 복사하고 bool_in2 를 BOR_02.In2 에 복사하며 모든 BOR_02 입력에 논리합 연산을 수행한 결과를 BOR_02.Out 에 저장한 다음 BOR_02.Out 을 value_result_or 에 복사합니다.

bool_in1 값:	bool_in2 값:	value_result_or 값:
0	0	0
0	1	1
1	0	1
1	1	1



FBD 펑션

추가 참조

비트별 논리합(OR) 페이지의 499

FBD 펑션 페이지의 472

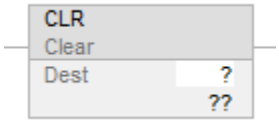
지우기(CLR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

CLR 명령어는 Dest 의 모든 비트를 해제합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

CLR 명령어는 기본 데이터 유형을 지원합니다. *기본 데이터 유형*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Dest	SINT INT DINT REAL	태그	해제할 태그.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. Dest 를 0 으로 해제합니다.
사후 스캔	N/A

예

래더 다이어그램

추가 참조

이동 명령어 페이지의 475

배열을 통한 인덱스 페이지의 978

기본 데이터 유형 페이지의 972

연산 상태 플래그 페이지의 963

마스킹된 이동(MVM)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

MVM 명령어는 Source 를 Destination 에 복사하여 데이터의 일부가 마스킹되도록 합니다.

MVM 명령어는 마스크를 사용하여 Source 데이터 비트를 통과시키거나 차단합니다. 마스크의 "1"은 데이터 비트가 통과되었음을 의미하고, 마스크의 "0"은 데이터 비트가 차단되었음을 나타냅니다.

정수 데이터 유형이 혼합된 경우 이 명령어가 작은 정수 데이터 유형의 상위 비트를 0 으로 채워 가장 큰 데이터 유형의 크기와 같아지게 합니다.

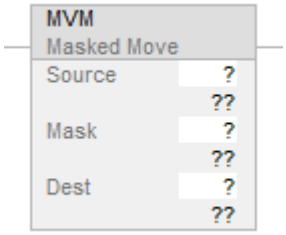
즉시 마스크 값 입력

마스크가 입력되면 프로그래밍 소프트웨어가 기본적으로 십진수 값으로 설정됩니다. 다른 형식을 사용하여 마스크를 입력하려면 값 앞에 알맞은 접두사를 붙입니다.

접두사	설명
16#	16 진수(예: 16#0F0F)
8#	8 진수(예: 8#16)
2#	2 진수(예: 2#00110011)

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Source	SINT INT DINT	즉시 태그	이동할 값
Mask	SINT INT DINT	즉시 태그	차단하거나 통과시킬 비트
Dest	SINT INT DINT	태그	결과를 저장하는 태그

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	아니요
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

컨트롤러	마이너 폴트 발생 조건:	폴트 유형	폴트 코드
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	기능이 활성화되고 오버플로가 감지됨	4	4
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	N/A	N/A	N/A

배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

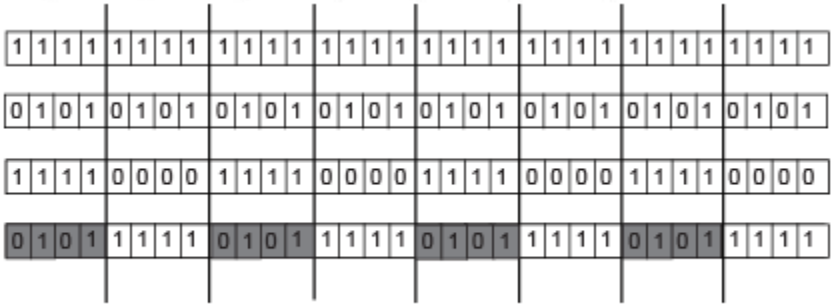
실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 Mask 를 통해 Source 를 통과시키고 결과를 Destination 에 복사합니다. Destination 의 마스킹되지 않은 비트는 그대로 유지됩니다.
사후 스캔	N/A

예

래더 다이어그램



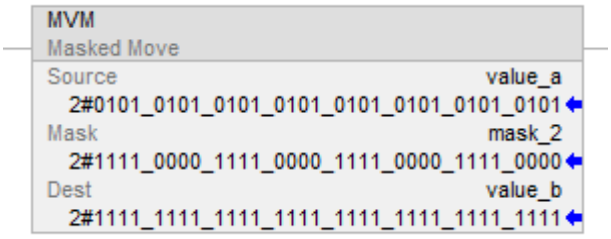
The shaded boxes show the bits that changed in value_b

행 1: MVM 전에 value_b

행 2: value_a

행 3: mask_2

행 4: MVM 후에 value_b



데이터를 value_a 에서 value_b 로 복사하여 데이터가 마스킹되도록 합니다(0 을 사용하여 value_a 의 데이터를 마스킹함).

추가 참조

이동 명령어 페이지의 475

데이터 변환 페이지의 967

배열을 통한 인덱스 페이지의 978

마스킹된 이동(대상 포함)(MVMT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

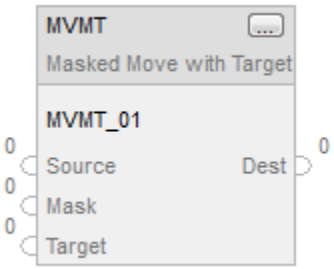
MVMT 명령어는 Source 를 Destination 에 복사하여 데이터의 일부가 마스킹되도록 합니다.

사용 가능한 언어

래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

평션 블록



ST(스트럭처드 텍스트)

MVMT(MVMT_tag);

피연산자

ST(스트럭처드 텍스트)

변수	유형	형식	설명
MVMT tag	FBD_MASKED_MOVE	Structure	MVMT 구조

ST(스트럭처드 텍스트) 내에서 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

평션 블록

피연산자	유형	형식	설명
MVMT tag	FBD_MASKED_MOVE	Structure	MVMT 구조

FBD_MASKED_MOVE 구조

입력 파라미터	데이터 유형	설명
활성화 입력	BOOL	선택하지 않으면 명령이 실행되지 않고 출력이 업데이트되지 않습니다. 설정되면 명령어가 실행됩니다. 기본값은 설정 상태입니다.
소스	DINT	Mask 의 값을 기반으로 Destination 으로 이동할 입력값. 유효값 = 모든 정수
마스크	DINT	Source 에서 Dest 로 이동할 비트의 마스크. 모든 비트를 1 로 설정하면 해당 비트가 Source 에서 Dest 로 이동합니다. 모든 비트를 0 으로 설정하면 해당 비트가 Source 에서 Dest 로 이동하지 않습니다. 유효값 = 모든 정수
Target	DINT	Mask 를 통해 Source 비트를 이동하기 전에 Dest 로 이동할 입력값. 유효값 = 모든 정수

출력 파라미터	데이터 유형	설명
활성화 출력(EnableOut)	BOOL	명령어의 활성화 여부를 나타냅니다.
Dest	DINT	마스킹된 이동 작업의 결과.

설명

활성화된 경우 MVMT 명령어는 마스크를 사용하여 Source 데이터 비트를 통과시키거나 차단합니다. 마스크의 "1"은 데이터 비트가 전달되었음을 의미합니다. 마스크의 "0"은 데이터 비트가 차단되었음을 의미합니다.

정수 데이터 유형을 혼합 사용하는 경우, 명령어는 작은 정수 데이터 유형이 가장 큰 데이터 유형과 크기가 같아질 때까지 작은 데이터 유형의 상위 비트를 0 으로 채웁니다.

입력 참조를 사용하여 즉시 마스크 값 입력

마스크를 입력하면 프로그래밍 소프트웨어가 기본적으로 십진수 값으로 설정됩니다. 다른 형식을 사용하여 마스크를 입력하려면 해당 값 앞에 올바른 접두어를 사용합니다.

접두사	설명
16#	16 진수(예: 16#0F0F)
8#	8 진수(예: 8#16)
2#	2 진수(예: 2#00110011)

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	아니요
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예(출력에 대해 영향 있음)

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

평션 블록

조건/상태	취해진 조치
사전 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 거짓임	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableIn 및 EnableOut 비트가 참으로 설정됩니다. 명령어가 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	EnableIn 및 EnableOut 비트가 거짓으로 해제됩니다.

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	펑션 블록 표의 사전 스캔을 참조하십시오.
정상 실행	펑션 블록 표에서 Tag.EnableIn 이 참임을 참조하십시오.
사후 스캔	펑션 블록 표에서 사후 스캔을 참조하십시오.

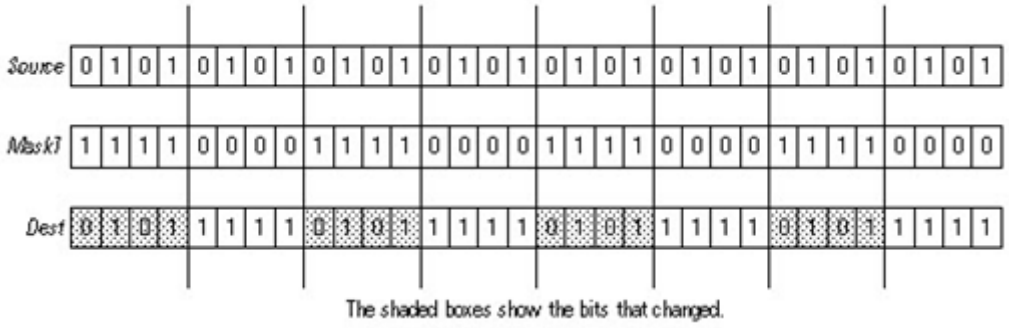
예제

1 단계

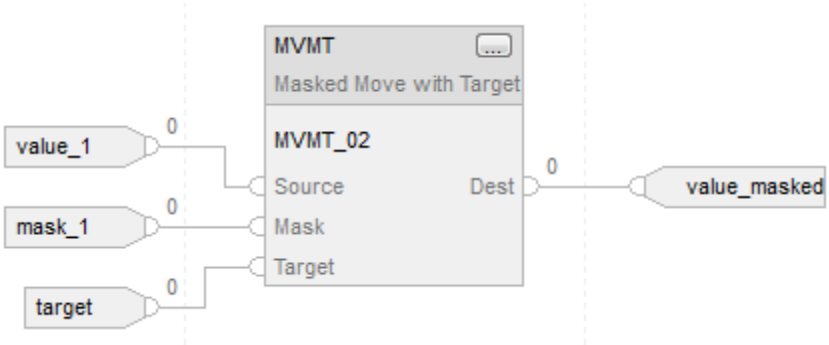
컨트롤러가 Target 을 Dest 에 복사합니다.

2 단계

명령어가 Source 를 마스킹하고 이를 Dest 과 비교합니다. Dest 에서 필요한 모든 내용을 변경하면 value_masked 의 입력 파라미터가 됩니다. Source 와 Target 은 그대로 유지됩니다. 마스크의 값이 0 이면 명령어가 해당 비트를 비교하지 못합니다.



평션 블록



ST(스트럭처드 텍스트)

```
MVMT_01.Source := value_1;
MVMT_01.Mask := mask_1;
MVMT_01.Target := target;

MVMT(MVMT_01);

value_masked := MVMT_01.Dest;
```

추가 참조

마스킹된 이동(MVM) 페이지의 527

데이터 변환 페이지의 967

ST(스트럭처드 텍스트) 구문 페이지의 997

공통 특성 페이지의 963

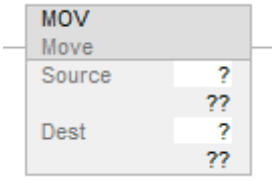
이동(MOV)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

MOV 명령어는 Source 의 복사본을 Dest 로 이동합니다. 소스는 변경되지 않습니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

팁: 동일한 결과를 얻으려면 식에 할당 ":="을 사용하십시오. ST(스트럭쳐드 텍스트) 내에서 식의 구문과 할당에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

숫자

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
소스	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	immediate 태그	이동할 값
Dest	SINT INT DINT REAL	SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL	태그	결과를 저장하는 태그

문자열(CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러의 경우만 해당)

피연산자	데이터 유형	형식	설명
소스	문자열 유형	immediate 태그	이동할 문자열
Dest	문자열 유형	태그	결과를 저장하는 태그

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

*연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

마이너 폴트 발생 조건:	폴트 유형	폴트 코드
오버플로 감지 기능이 활성화되어 있고 소스 갑시 Dest 유형의 범위에서 벗어났습니다.	4	4

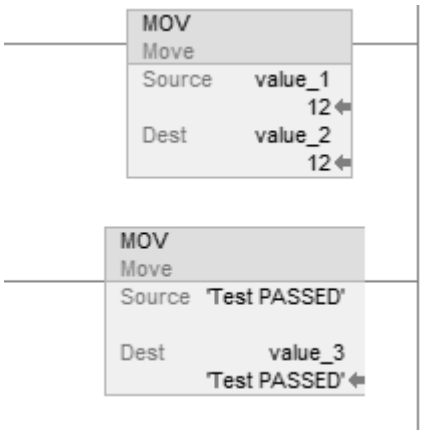
실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. 명령어가 Source 를 Dest 에 복사합니다. **문자열 피연산자:** Source.LEN > SIZE(Dest.DATA)인 경우 문자열이 길이에 맞춰 잘립니다. S:V 가 설정됩니다.
사후 스캔	N/A

예제

래더 다이어그램



ST(스트럭쳐드 텍스트)

```
value_2 := value_1;

value_3 := 'Test PASSED';
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

연산 상태 플래그 페이지의 963

배열을 통한 인덱스 페이지의 978

이동 명령어 페이지의 475

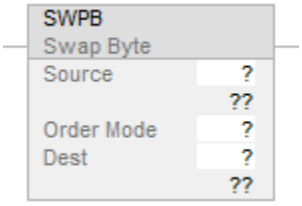
바이트 스왑(SWPB)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

SWPB 명령어는 Source 의 바이트 순서를 다시 배열합니다. 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

SWPB(Source, Order Mode, Dest);

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	데이터 유형	형식	설명
소스	INT DINT	태그	다시 배열할 비트를 포함한 태그.
Order Mode		목록 항	이 피연산자는 다시 정렬할 방법을 지정합니다. 순서 모드 표를 참조하십시오.
Dest	INT DINT	태그	바이트를 새로운 순서로 저장하는 태그. Dest 표를 참조하십시오.

HIGH/LOW 순서 모드를 선택한 경우 HIGHLOW(슬래시 없음)로 입력하십시오. ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

Order Mode

Source 가 다음인 경우	바이트 변경할 패턴(각 문자는 다른 유형을 나타냄)	다음을 선택함
INT	AB => BA	모든 옵션
DINT	ABCD => DCBA	REVERSE
	ABCD => CDAB	WORD
	ABCD => BADC	HIGH/LOW

Dest

Source 가 다음인 경우	Destination 이 다음임
INT	INT, DINT 대상이 DINT 이면 바이트 스왑 후에 결과가 부호 확장됩니다.
DINT	DINT

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 지정된 바이트를 다시 배열합니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

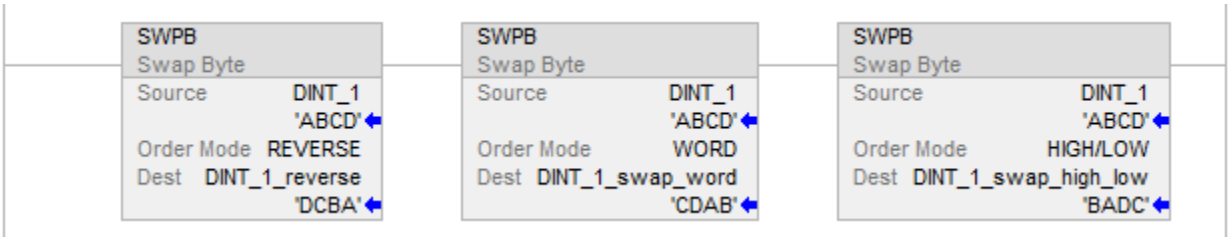
조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 참조하십시오.

예제

예 1 - DINT 태그의 바이트 스왑

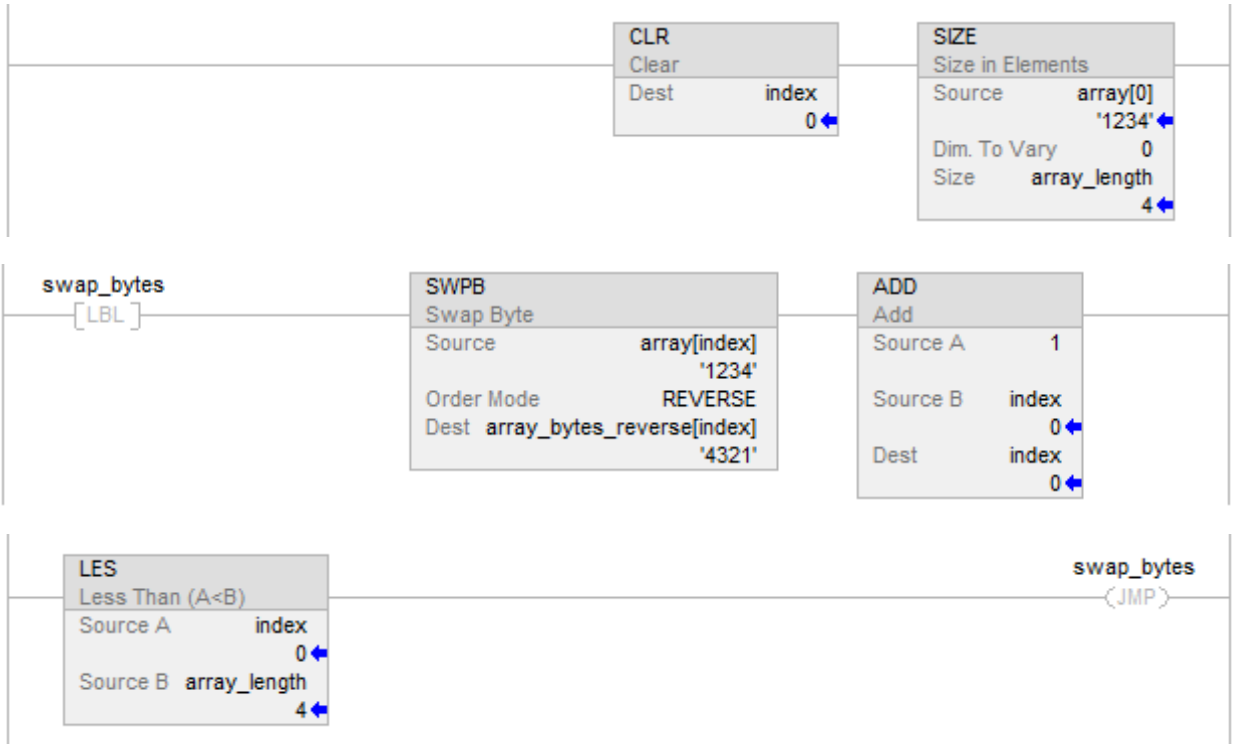
3 개의 SWPB 명령어가 다른 순서 모드에 따라 DINT_1 의 바이트를 다시 정렬합니다. 디스플레이 스타일은 ASCII 이며 각 문자는 1 바이트를 나타냅니다. 모든 명령어가 바이트를 새로운 순서로 다른 Destination 에 저장합니다.

래더 다이어그램



예 2 - 배열의 모든 요소의 바이트 스왑

래더 다이어그램



예 3: ST(스트럭처드 텍스트)의 SWPB

ST(스트럭처드 텍스트)

```
index := 0;
SIZE (array[0],0,array_length);
REPEAT
  SWPB(array[index],REVERSE,array_bytes_reverse[index]);
  index := index + 1;
UNTIL(index >= array_length)END_REPEAT;
```

추가 참조

이동 명령어 페이지의 475

배열을 통한 인덱스 페이지의 978

데이터 변환 페이지의 967

ST(스트럭쳐드 텍스트) 구문 페이지의 997

배열(파일)/기타 명령어

배열(파일)/기타 명령어

파일/기타 명령어는 데이터의 배열에서 작동됩니다.

사용 가능한 명령어

래더 다이어그램

FAL	FSC	COP	CPS	FLL	AVE

평션 블록

사용할 수 없음

ST(스트럭처드 텍스트)

SIZE	FSC	COP	CPS

실행할 작업:	사용할 명령어
배열의 값에 대해 산술, 로직, 이동 및 함수 연산을 수행합니다.	FAL
배열의 값을 검색하고 비교합니다.	FSC
한 배열의 내용을 다른 배열에 복사합니다.	COP
Source 의 값을 Destination 에 복사합니다.	CPS
배열을 특정 데이터로 채웁니다.	FLL
값 배열의 평균값을 계산합니다.	AVE
배열 데이터의 한 차원을 오름차순으로 정렬합니다.	SRT
값 배열의 표준 편차를 계산합니다.	STD
배열의 한 차원의 크기를 계산합니다.	SIZE

데이터 유형을 혼합할 수 있지만 정확도 손실 및 반올림 에러가 발생할 수 있으며 명령어를 실행하는 데 시간이 더 걸립니다. 결과가 잘렸는지 확인하려면 S:V 비트를 확인하십시오.

볼드체 데이터 유형은 최적의 데이터 유형을 나타냅니다. 명령어의 모든 피연산자가 동일한 최적의 데이터 유형(일반적으로 DINT 또는 REAL)을 사용하는 경우 명령어는 가장 적은 메모리를 사용하면서 보다 빠르게 실행됩니다.

연산 모드 선택

FAL 및 FSC 명령어의 경우 이 모드는 컨트롤러에 배열 연산을 분배하는 방법을 지시합니다.

실행할 작업:	선택할 모드:
다음 명령어를 계속하기 전에 배열에서 지정된 모든 요소에 대해 연산을 수행합니다.	전체 모드
배열 연산을 여러 스캔에 분배합니다. 스캔당 연산 작업을 수행할 요소의 수(1 ~ 2147483647)를 입력합니다.	숫자 모드
렁-입력-조건이 거짓에서 참으로 바뀔 때마다 배열 내 요소 하나를 조작합니다.	증가 모드

추가 참조

전체 모드 페이지의 611

숫자 모드 페이지의 612

증가 모드 페이지의 615

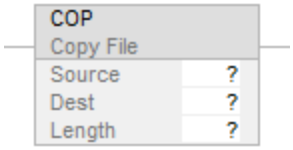
파일 복사(COP), 동기식 파일 복사(CPS)

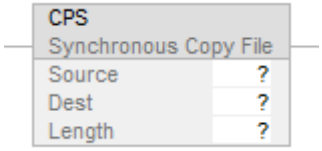
이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

COP 및 CPS 명령어는 Source 의 값을 Dest 의 값으로 복사합니다. 소스는 변경되지 않습니다.

사용 가능한 언어

래더 다이어그램





펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

COP(Source,Dest,Length);

CPS(Source,Dest,Length);

피연산자

중요:	다음과 같은 경우 작업 시 예외가 발생할 수 있습니다. • 출력 태그 피연산자가 덮어써집니다. • 구조 피연산자의 구성원이 덮어써집니다. • 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Source	SINT INT DINT LINT REAL 문자열 유형 구조	태그	복사할 초기 요소
Dest	SINT INT DINT LINT REAL 문자열 유형 구조	태그	소스가 덮어쓸 초기 요소
Length	SINT INT DINT	즉시 태그	복사할 대상 요소의 수

ST(스트럭처드 텍스트)

피연산자	데이터 유형	형식	설명
Source	SINT INT DINT LINT REAL 문자열 유형 구조	태그	복사할 초기 요소
Dest	SINT INT DINT LINT REAL 문자열 유형 구조	태그	소스가 덮어쓸 초기 요소
Length	SINT INT DINT	즉시 태그	복사할 대상 요소의 수

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. 명령어가 데이터를 복사합니다.
사후 스캔	N/A

ST(스트럭쳐드 텍스트)

조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 확인하십시오.

COP 와 CPS 명령어의 실행 중에 다른 컨트롤러 조치가 복사 작업을 중단하면서 소스를 변경하려고 시도할 수 있습니다.

소스 또는 대상:	원하는 작업:	선택할 명령어:	참고:
• 생산된 태그 • 사용된 태그 • I/O 데이터 • 다른 태스크가 덮어쓸 수 있는 데이터	복사 작업 중에 소스 데이터가 변경되지 않게 합니다	CPS	CPS 명령어를 중단하려고 시도하는 태스크가 명령어가 완료될 때까지 지연됩니다. CPS 명령어의 실행 시간을 추정하려면 ControlLogix System User Manual(발행 번호: 1756-UM001)을 참조하십시오.
	복사 작업 중에 소스 데이터의 변경을 허용합니다.	COP	
위의 해당 사항 없음	---------->	COP	

COP 와 CPS 명령어는 연속 메모리에 작업하여 직접적인 바이트간 메모리 복사를 수행합니다.

Source 와 Dest 의 데이터 유형이 다른 경우, 복사되는 바이트 수는 다음 중에서 작은 수와 동일합니다.

- 요청된 수량은 길이 x 와 같음(대상 요소의 바이트 수)
- 대상 태그의 바이트 수
- Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580 또는 GuardLogix 5580 컨트롤러의 경우, 소스 태그의 바이트 수

팁: 대상 또는 소스 태그의 끝은 기본 태그의 마지막 바이트로 정의합니다. 태그가 구조인 경우, 태그의 끝은 구조의 마지막 요소의 마지막 바이트입니다. 이는 COP 와 CPS 명령어가 구성원 배열의 끝을 지나서 작성할 수 있지만 어떠한 경우에도 기본 태그의 끝을 지나서 작성하지 않음을 의미합니다.

중요: 명령어가 변경되어서는 안 되는 데이터를 변경하지 않는지 테스트하고 확인하십시오.

예

예 1

배열 복사하기.

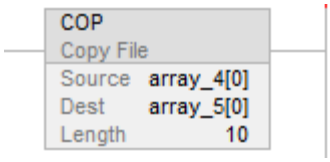
활성화된 경우, COP 명령어는 array_4 에서 array_5 로 40 바이트를 복사합니다.

array_4 는 DINT(요소 당 4 바이트)이고, 10 요소(총 크기 = 40 바이트)로 구성됩니다.

array_5 는 DINT(요소 당 4 바이트)이고, 10 요소(총 크기 = 40 바이트)로 구성됩니다.

Length 가 10 개 대상 요소의 복사를 지시하므로 40 바이트가 복사됩니다.

래더 다이어그램



ST(스트럭처드 텍스트)

COP(array_4[0],array_5[0],10);

예 2

구조를 복사합니다.

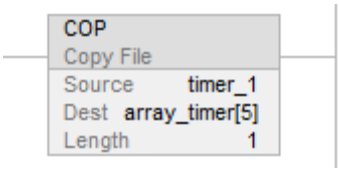
활성화된 경우, COP 명령어는 구조 timer_1 을 array_timer 의 요소 5 로 복사합니다.

timer_1 은 TIMER(총 크기 = 12 바이트)입니다..

array_timer 는 TIMER(요소 당 12 바이트)이고, 10 요소(총 크기 = 120 바이트)로 구성됩니다.

Length 가 1 대상 요소를 지시하므로 12 바이트가 복사됩니다.

래더 다이어그램



ST(스트럭처드 텍스트)

```
COP(timer_1,array_timer[5],1);
```

예 3

복사가 완료될 때까지 데이터의 변경을 방지하면서 배열 데이터를 복사합니다.

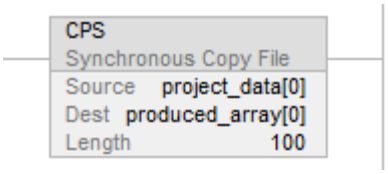
project_data 배열(100 요소)은 응용 프로그램에서 상이한 시점에 변경되는 다양한 값을 저장합니다. 한 인스턴스의 project_data 의 전체 이미지를 제 시간에 다른 컨트롤러로 전송하기 위해 CPS 명령어는 project_data 를 produced_array 로 복사합니다. CPS 명령어가 데이터를 복사하는 동안에는 어떤 I/O 업데이트나 기타 태스크도 데이터를 변경할 수 없습니다. produced_array 태그는 다른 컨트롤러들이 사용할 수 있도록 데이터를 ControlNet 네트워크에서 생산합니다.

project_data 는 DINT(요소당 4 바이트)이고, 100 요소(총 크기 = 400 바이트)로 구성됩니다.

produced_array 은 DINT(요소 당 4 바이트)이고, 100 요소(총 크기 = 400 바이트)로 구성됩니다.

Length 가 100 개 대상 요소를 지시하므로 400 바이트가 복사됩니다.

래더 다이어그램



ST(스트럭처드 텍스트)

```
CPS(project_data[0],produced_array[0],100);
```

예 4

복사가 완료될 때까지 데이터의 전송을 방지하면서 데이터를 생산된 태그로 복사합니다.

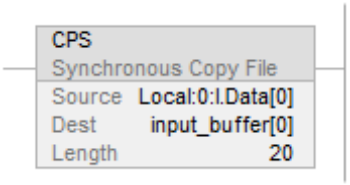
Local:0:I.Data 가 슬롯 0 에서 1756-DNB 모듈에 연결된 DeviceNet 네트워크에 입력 데이터를 저장합니다. 입력을 응용 프로그램과 동기화하기 위해 CPS 명령어는 입력 데이터를 input_buffer 로 복사합니다. CPS 명령어가 데이터를 복사하는 동안에는 어떤 I/O 업데이트도 데이터를 변경할 수 없습니다. 응용 프로그램이 실행되면 input_buffer 에 있는 입력 데이터를 입력 항목으로 사용합니다.

Local:O:I.Data 는 DINT(요소당 4 바이트)이고, 2 요소(총 크기 = 8 바이트)로 구성됩니다.

input_buffer 는 DINT(요소당 4 바이트)이고, 20 요소(총 크기 = 80 바이트)로 구성됩니다.

Length 가 20 개 대상 요소의 복사를 지시합니다(4 X 20 = 80 바이트). 그러나 소스는 8 바이트만 제공할 수 있으므로 8 바이트가 복사됩니다.

래더 다이어그램

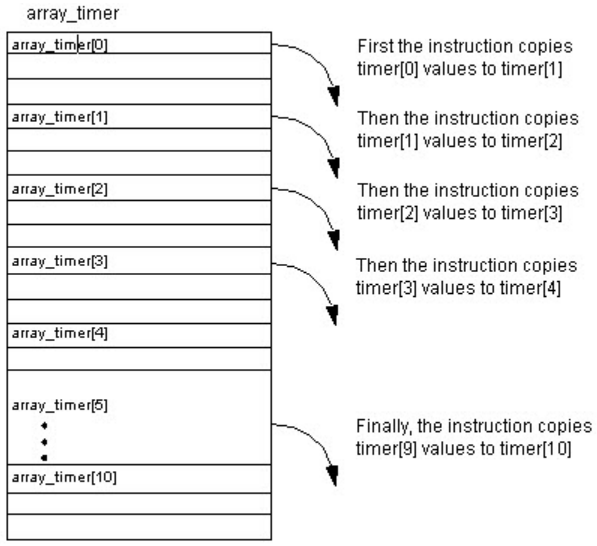


ST(스트럭처드 텍스트)

```
CPS(Local:0:I.Data[0], input_buffer[0], 20);
```

예 5

배열 구조를 초기화하고, 최초의 요소를 초기화하며, COP 를 사용하여 이를 배열의 나머지 부분에 복사합니다.

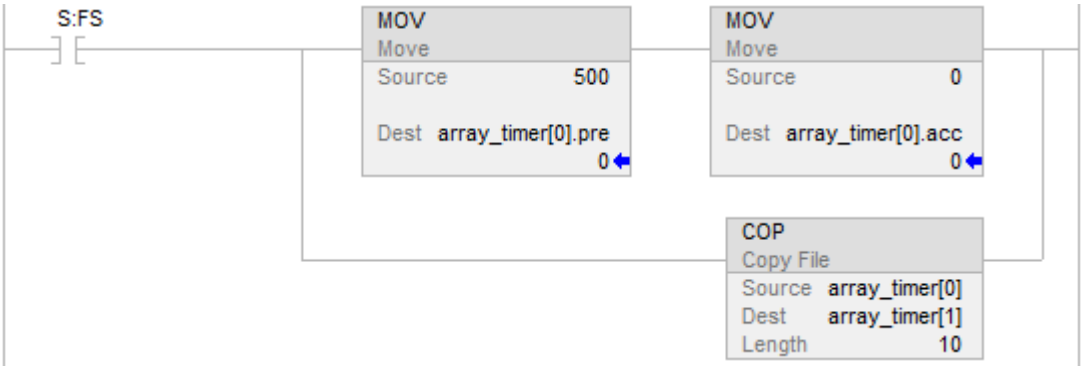


이 예에서는 배열 또는 타이머 구조를 초기화합니다. 활성화된 경우, MOV 명령어는 최초의 array_timer 요소의 .PRE 및 .ACC 값을 초기화합니다. 활성화된 경우, COP 명령어는 array_timer[0]부터 시작하여 연속적인 바이트 블록을 복사합니다. 길이는 9 타이머 구조입니다.

array_timer 는 TIMER(요소당 12 바이트)이고, 15 요소(총 크기 = 180 바이트)로 구성됩니다.

Length 가 10 개 대상 요소를 지시하므로 120 바이트가 복사됩니다.

래더 다이어그램



ST(스트럭처드 텍스트)

```
IF S:FS THEN

array_timer[0].pre := 500;

array_timer[0].acc := 0;

COP(array_timer[0],array_timer[1],10);

END_IF;
```

예 6

다른 크기의 배열 복사하기.

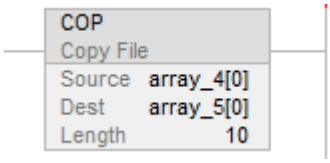
활성화된 경우, COP 명령어는 SINT array_6 의 바이트를 DNT array_7 로 복사합니다.

array_6 은 SINT(요소당 1 바이트)이고, 5 요소(총 크기 = 5 바이트)로 구성됩니다.

array_7 은 DINT(요소당 4 바이트)이고, 10 요소(총 크기 = 40 바이트)로 구성됩니다.

Length 가 20 개 대상 요소의 복사를 지시합니다(4 X 20 = 80 바이트). 그러나 대상은 40 바이트만 수용하고 소스는 5 바이트만 제공할 수 있으므로 5 바이트가 복사됩니다.

래더 다이어그램



ST(스트럭처드 텍스트)

```
COP(array_4[0],array_5[0],10);
```

추가 참조

배열을 통한 인덱스 페이지의 978

파일/기타 명령어 페이지의 545

이동/로직 명령어 페이지의 475

ST(스트럭쳐드 텍스트) 구문 페이지의 997

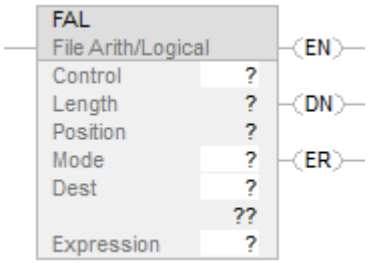
파일 산술 및 로직(FAL)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

FAL 명령어는 배열에 저장된 데이터에 대한 복사, 산술, 로직 및 함수 연산을 수행합니다. FAL 명령어의 렁-입력-조건이 거짓에서 참으로 전환되면 지정된 식이 지정된 반복 모드로 실행됩니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형	형식	설명
제어	CONTROL	태그	연산의 제어 구조
길이(Length)	DINT	즉시	조작할 배열 내 요소의 수
위치(Position)	DINT	즉시	배열로 오프셋 초기값은 일반적으로 0 입니다.
Mode	DINT	즉시	연산 배포 방법을 보여줍니다. INC, ALL 을 선택하거나, 1 ~ 2147483647 범위의 수를 입력함
Expression	SINT INT DINT REAL	즉시 태그	태그 및/또는 연산자로 구분된 즉시 값으로 구성된 식.
Destination	SINT INT DINT REAL	태그	식의 값이 대상에 저장됩니다.

CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 FAL 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 마지막 요소에서 작업한 경우에 설정됩니다(.POS = .LEN).
.ER	BOOL	오버플로가 발생하면 두 플랫폼 모두 .ER 을 설정하고 명령어 실행을 중지합니다. 다음 컨트롤러가 오버플로를 생성합니다. • CompactLogix 5370 • ControlLogix 5570
.LEN	DINT	길이는 FAL 명령어가 작동되는 배열의 요소 수를 지정합니다.
.POS	DINT	위치에는 명령어가 현재 액세스하고 있는 요소의 위치가 표시됩니다.

ST(스트럭쳐드 텍스트) 내에서 식의 구문에 대한 자세한 내용은 ST(스트럭쳐드 텍스트) 구문[2]을 참조하십시오.

식의 값은 지정된 대상 태그에 저장됩니다. 오버플로가 발생하면 ER 비트를 설정하고 실행을 중지합니다. FAL 이 구성된 반복을 모두 완료하면 .DN 비트가 설정됩니다.

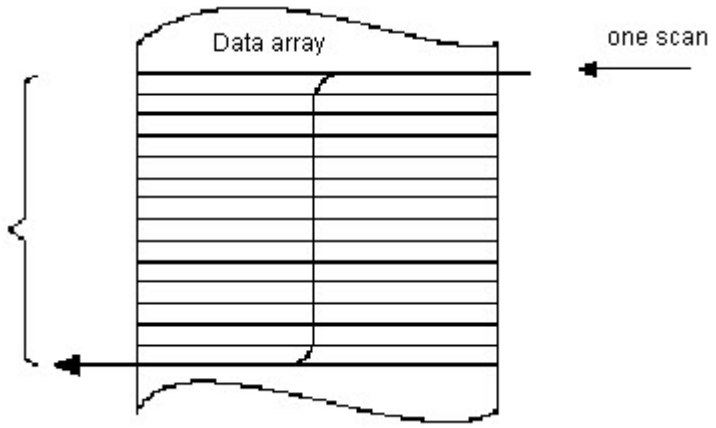
연산 모드 선택하기

FAL 명령어의 경우 이 모드가 컨트롤러에 배열 연산 배포 방법을 지시합니다.

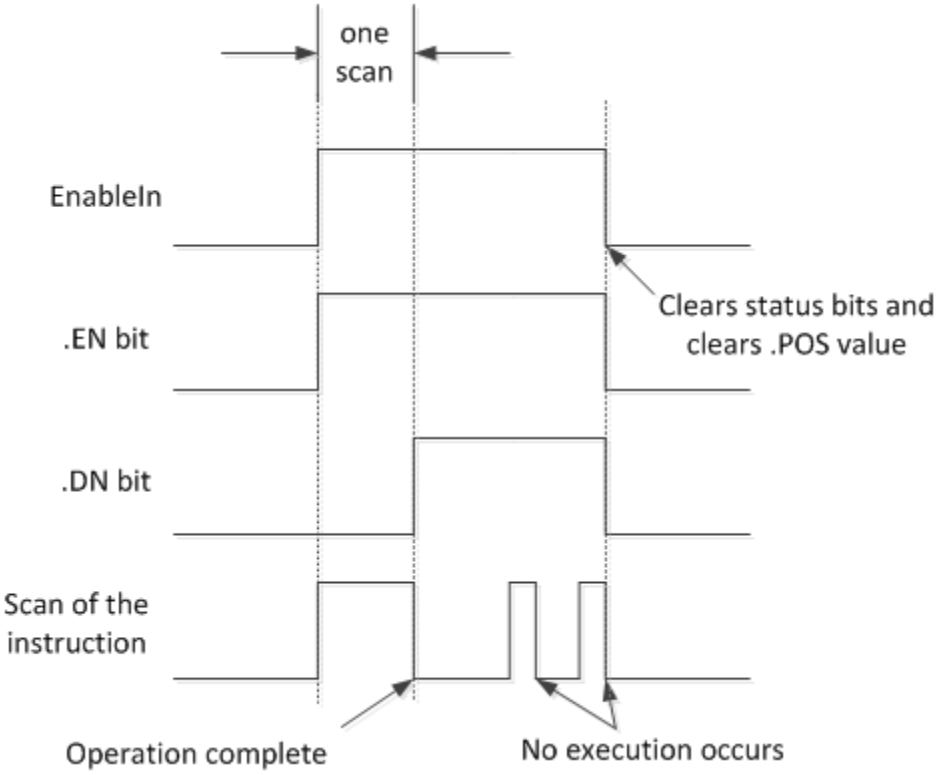
다음의 경우:	선택할 모드:
다음 명령어를 계속하기 전에 배열에서 지정된 모든 요소에 대한 연산 수행.	모두
여러 스캔을 통해 배열 연산 배포. 스캔당 작업할 요소의 수(1 ~ 2147483647)를 입력합니다.	숫자
EnableIn 이 거짓에서 참으로 전환될 때마다 배열의 한 요소 조작.	증가

전체 모드

전체 모드에서는 명령어가 다음 명령어를 계속하기 전에 배열의 지정된 모든 요소에 대한 연산을 수행합니다. 작업은 명령어의 EnableIn 이 거짓에서 참으로 바뀔 때 시작합니다. 제어 구조의 위치(.POS) 값은 명령어가 현재 사용하고 있는 배열 내 요소를 가리킵니다. .POS 값이 .LEN 값과 같아지거나 그보다 커지고, 식에서 오버플로가 발생하고 .ER 비트가 참으로 설정되면 연산이 중지됩니다.



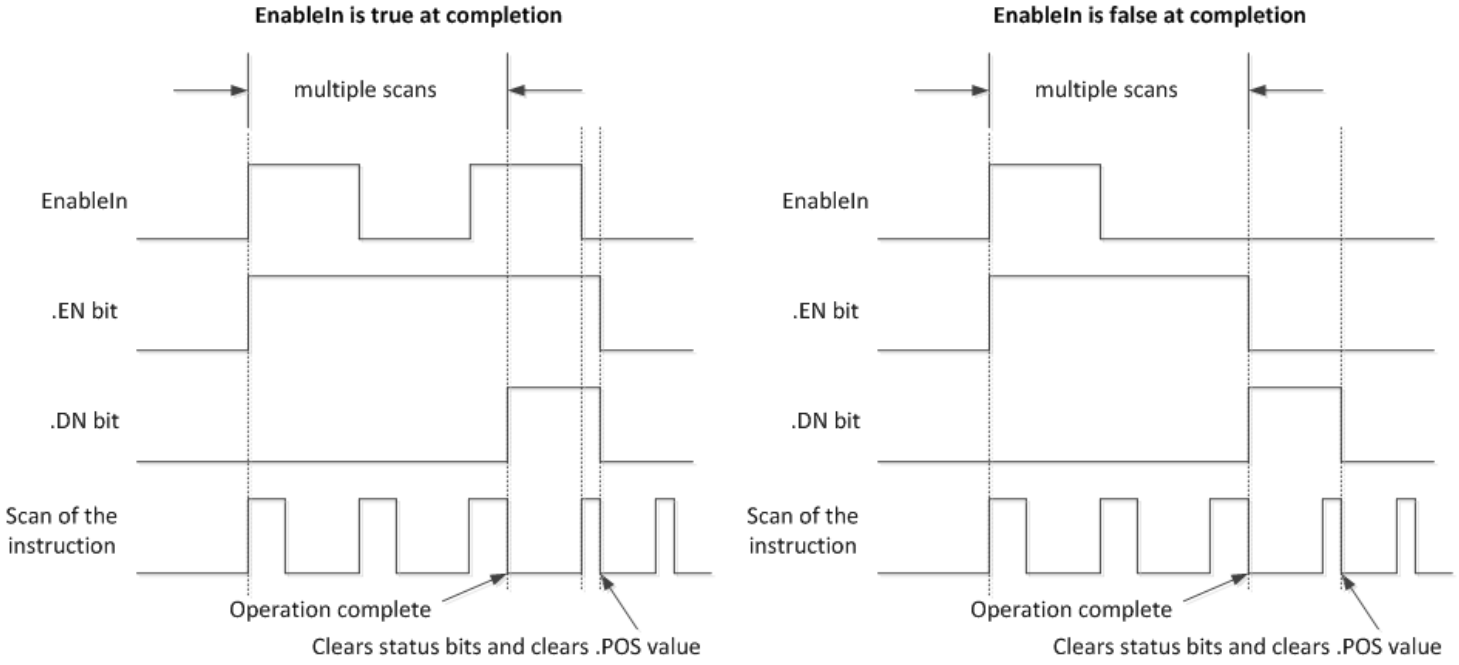
아래 타이밍 다이어그램에서는 상태 비트와 명령어 작업의 관계를 보여줍니다. 명령어 실행이 완료되면 .DN 비트가 참입니다. EnableIn 이 거짓이면 .DN 비트, .EN 비트, .POS 값이 지워집니다. 이 경우에만 EnableIn 거짓 - 참 전환에서 명령어의 다른 실행이 트리거될 수 있습니다.



숫자 모드

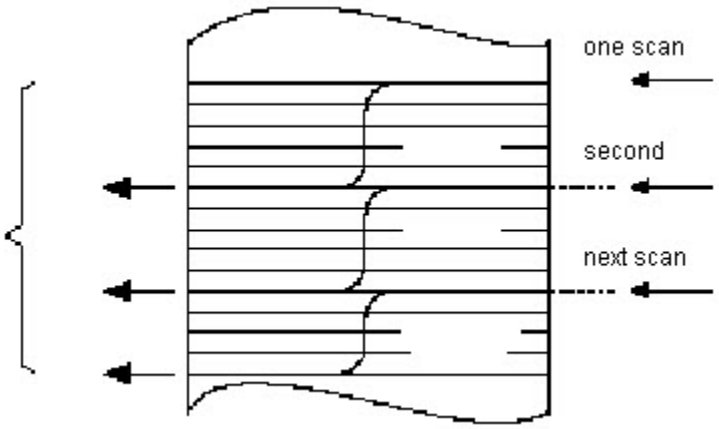
숫자 모드는 배열 작업을 여러 스캔 작업에 분배합니다. 비시간 임계 데이터나 대량의 데이터를 처리할 경우 이 모드를 사용합니다. 각 스캔에서 연산을 수행할 요소 수를 입력하면 스캔 시간이 더 짧게 유지됩니다.

EnableIn 이 거짓에서 참으로 전환되면 실행이 트리거됩니다. 이 경우 명령어는 전체 배열의 작업을 완료하는 데 필요한 스캔 작업의 횟수 만큼 스캔될 때마다 실행됩니다. 트리거되면 EnableIn 은 명령어의 실행을 중단하지 않고도 반복해서 변경할 수 있습니다.



.DN 비트가 설정될 때까지 숫자 모드에서 작동되는 파일 명령어의 결과를 사용하지 마십시오.

아래 타이밍 다이어그램에서는 상태 비트와 명령어 작업의 관계를 보여줍니다. 명령어 실행이 완료되면 .DN 비트가 설정됩니다.

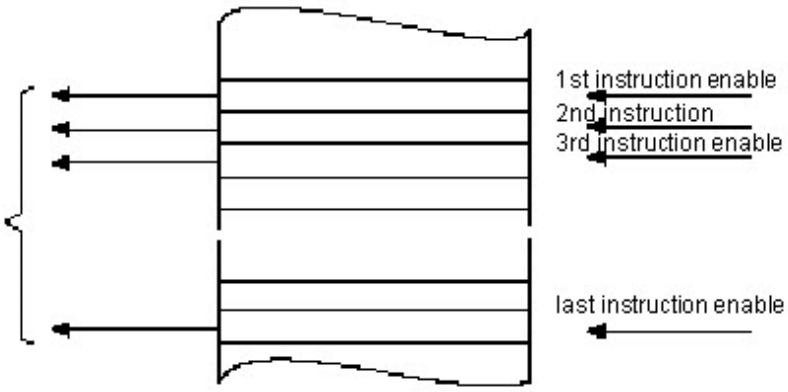


완료 시에 EnableIn 이 참이면 .EN 과 .DN 비트는 EnableIn 이 거짓으로 바뀔 때까지 참입니다. EnableIn 이 거짓으로 바뀌면 이들 비트와 .POS 값은 지워집니다.

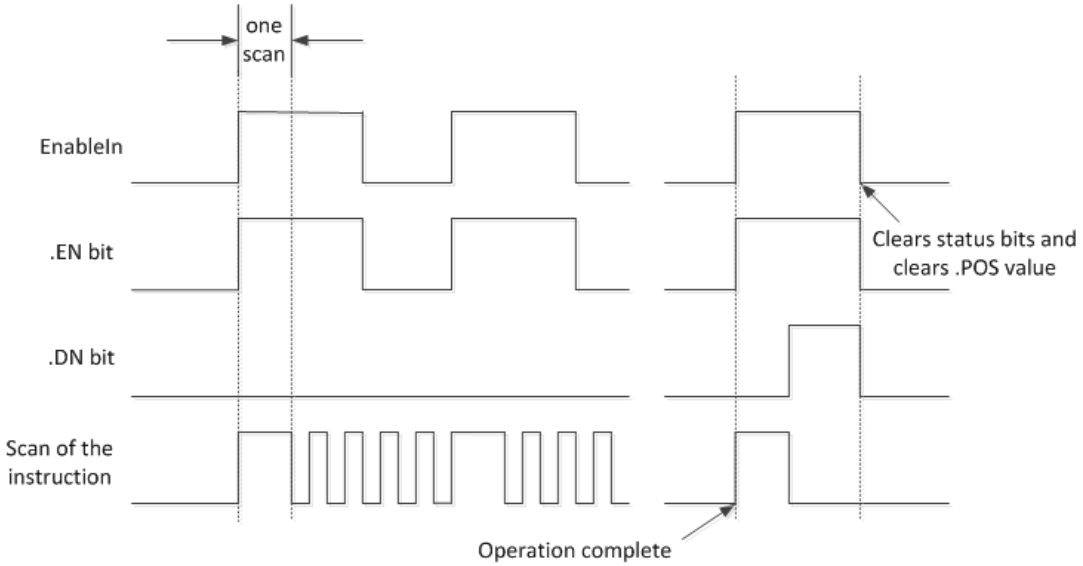
완료 시에 EnableIn 이 거짓이면 .EN 비트는 즉시 지워집니다. .DN 비트와 .POS 값은 .EN 비트가 지워진 직후의 스캔 작업 시에 지워집니다.

증가 모드

증가 모드는 명령어의 EnableIn 이 거짓에서 참으로 바뀔 때마다 배열 내 요소 하나를 조작합니다.



아래 타이밍 다이어그램에서는 상태 비트와 명령어 작업의 관계를 보여줍니다. 실행은 EnableIn 이 거짓에서 참으로 바뀌는 스캔에서만 진행됩니다. 이 상황이 발생할 때마다 배열 내 요소 하나만 조작됩니다. 두 번 이상의 스캔 동안 EnableIn 이 참으로 유지되면 명령어가 첫 번째 스캔 중에만 실행됩니다



EnableIn 이 참이면 .EN 비트가 설정됩니다. 배열 내 마지막 요소가 조작되면 .DN 비트가 설정됩니다. 마지막 요소가 조작되고 EnableIn 이 거짓으로 전환되면 .EN 비트, .DN 비트, .POS 값이 해제됩니다.

스캔당 요소 하나의 속도에 따른 증가 모드와 숫자 모드의 차이점은 다음과 같습니다.

스캔당 요소가 몇 개 있는 숫자 모드의 실행을 시작하려면 EnableIn 의 거짓 - 참 전환이 하나 있으면 됩니다. 명령어는 EnableIn 의 상태에 상관 없이 완료될 때까지 스캔마다 지정된 개수의 요소를 계속 실행합니다.

증가 모드는 EnableIn 이 거짓에서 참으로 전환되어야 배열 내 요소 하나를 작업할 수 있습니다.

포맷 식

식에서 사용하는 각 연산자에는 1 ~ 2 개의 피연산자(태그 또는 즉시 값)를 제공해야 합니다. 다음 표를 사용하여 식 내의 연산자 및 피연산자 형식을 지정합니다.

연산자 작용 대상:	사용 형식:	예
피연산자 1 개	연산자(피연산자)	ABS(tag)
피연산자 2 개	operand_a 연산자 operand_b	tag_b + 5 tag_c AND tag_d (tag_e**2) MOD (tag_f / tag_g)

연산 순서 정의

명령어는 식 안에 써넣은 연산을 작성 순서가 아닌 미리 규정된 순서에 따라 수행합니다. 작업 순서를 오버라이드하려면 해당 항을 모아 괄호 안에 넣음으로써 명령어가 다른 연선보다 우선 괄호 안의 연선을 수행하게 하면 됩니다.

연산 순서가 같은 경우에는 왼쪽에서 오른쪽 순으로 수행됩니다.

순서	연산
1	( )
2	ABS, ACS, ASN, ATN, COS, DEG, FRD, LN, LOG, RAD, SIN, SQR, TAN, TOD, TRN
3	**
4	- (부정), NOT
	*, /, MOD
6	- (빼기), +
7	논리곱
8	XOR
9	또는

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	아니요
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
.POS < 0 또는 .LEN < 0	4	21

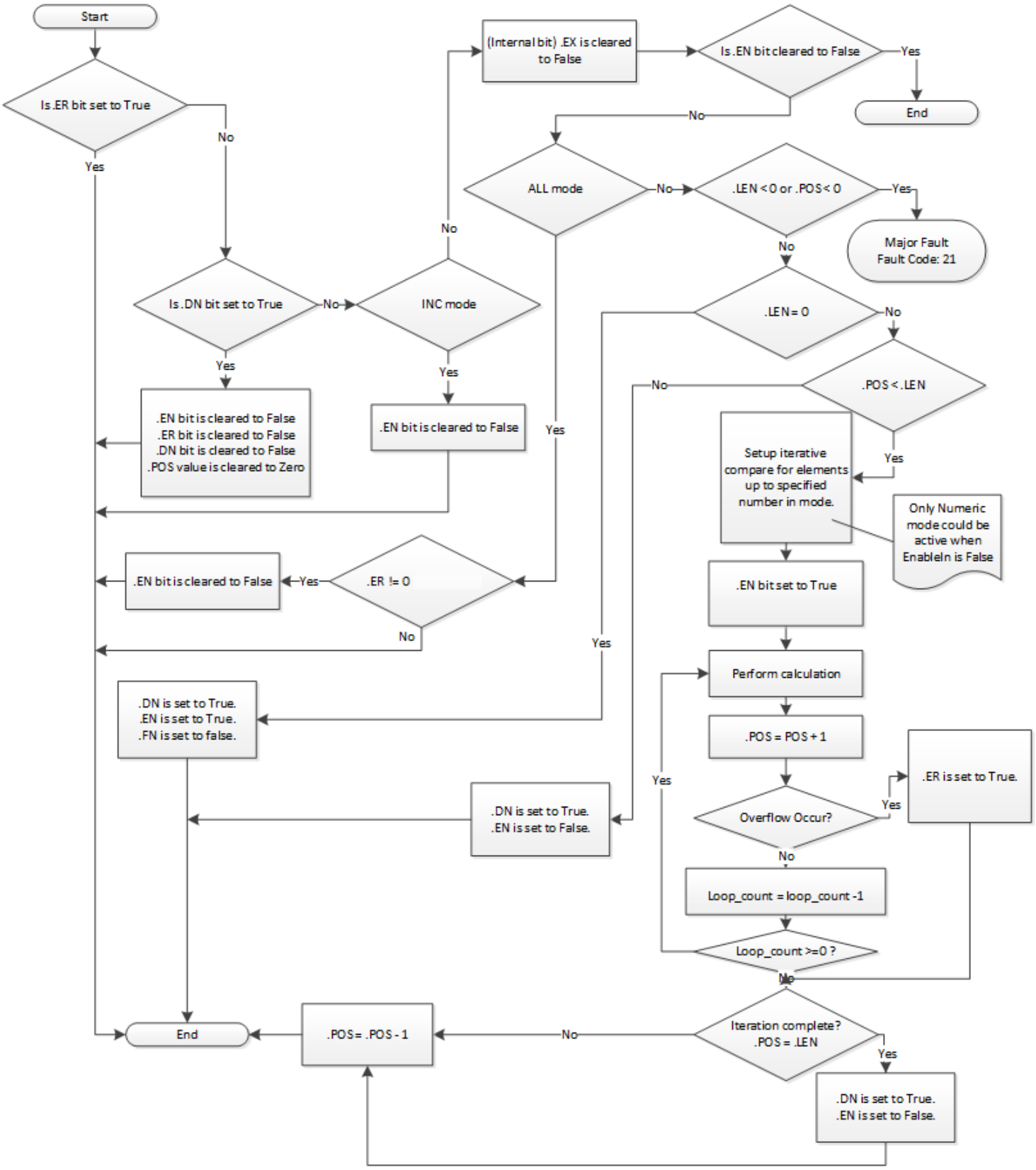
배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

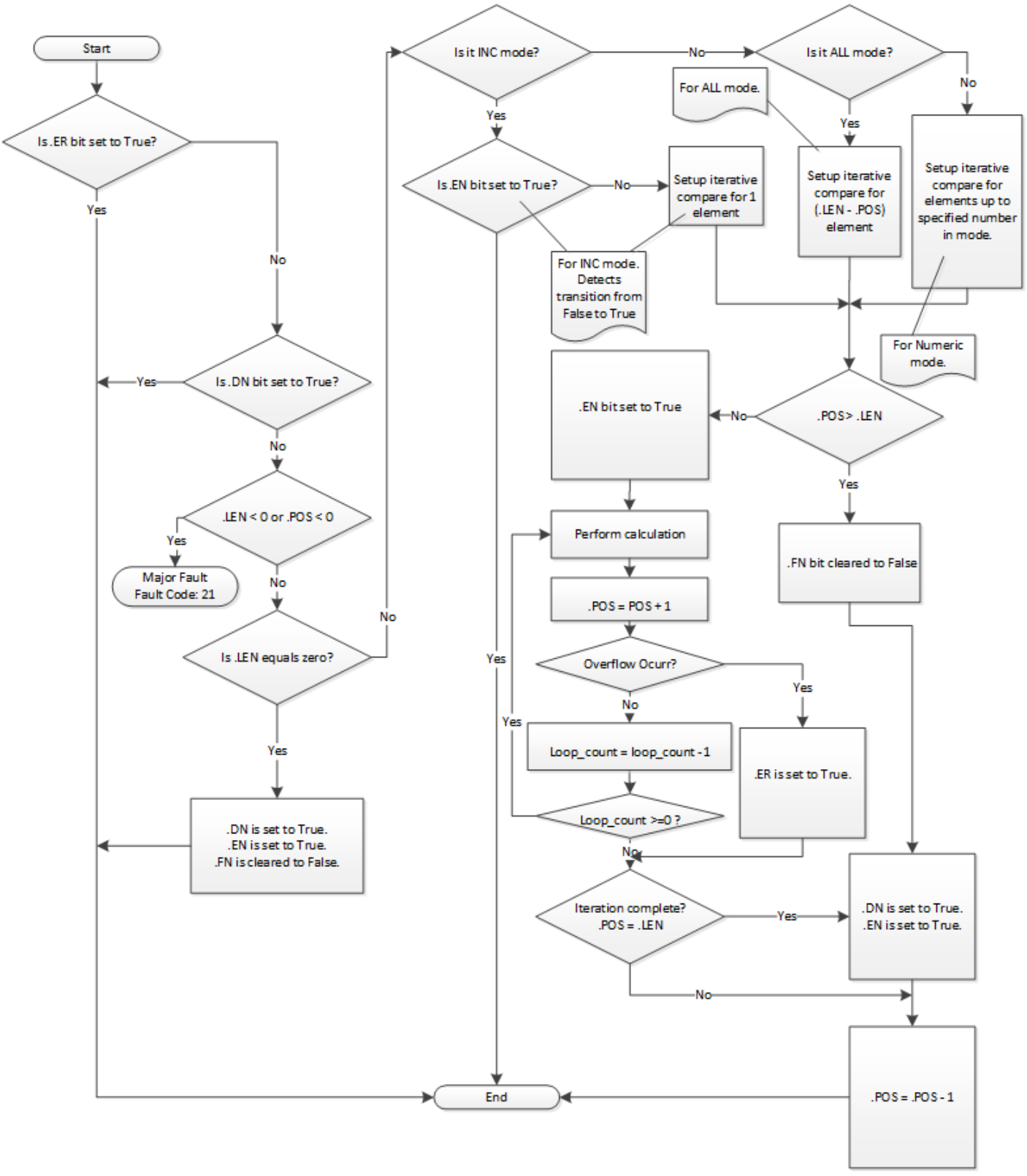
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	FAL 흐름도(렁-출력-조건이 거짓인 경우) 참조하십시오.
렁-입력-조건이 참임	FAL 흐름도(렁-출력-조건이 참인 경우) 참조하십시오.
사후 스캔	N/A

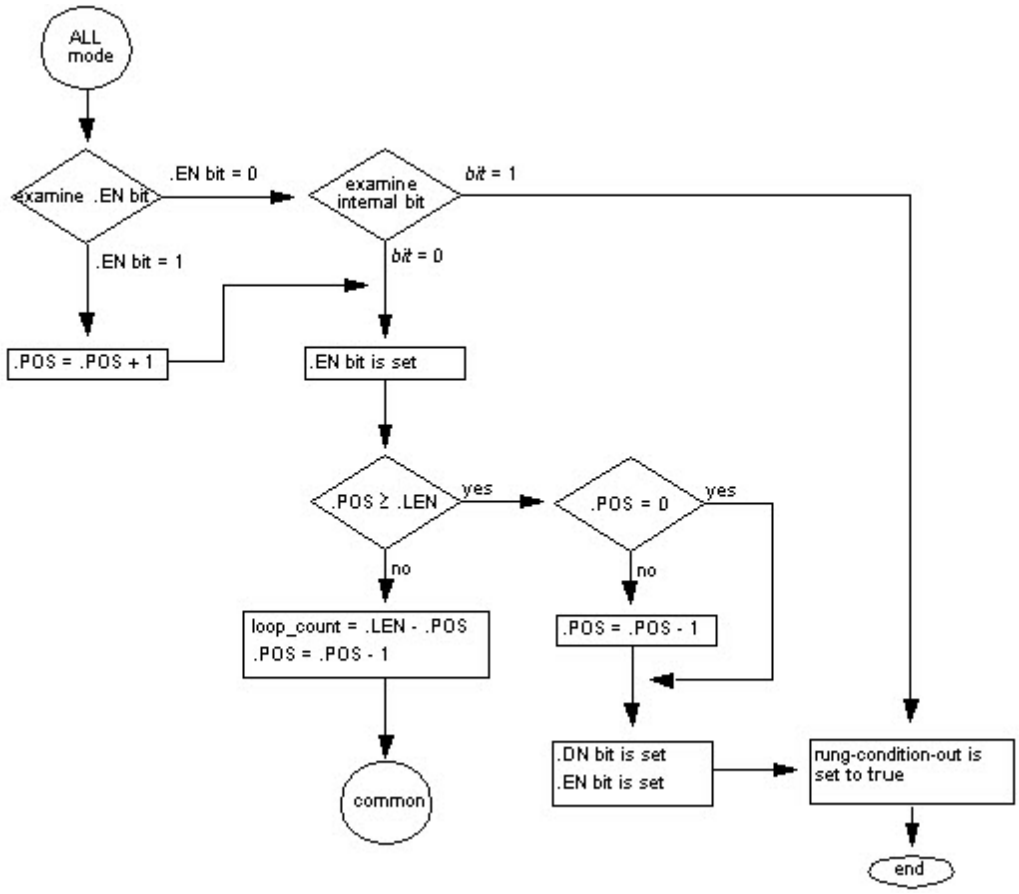
FAL 흐름도(렁-출력-조건이 거짓인 경우)



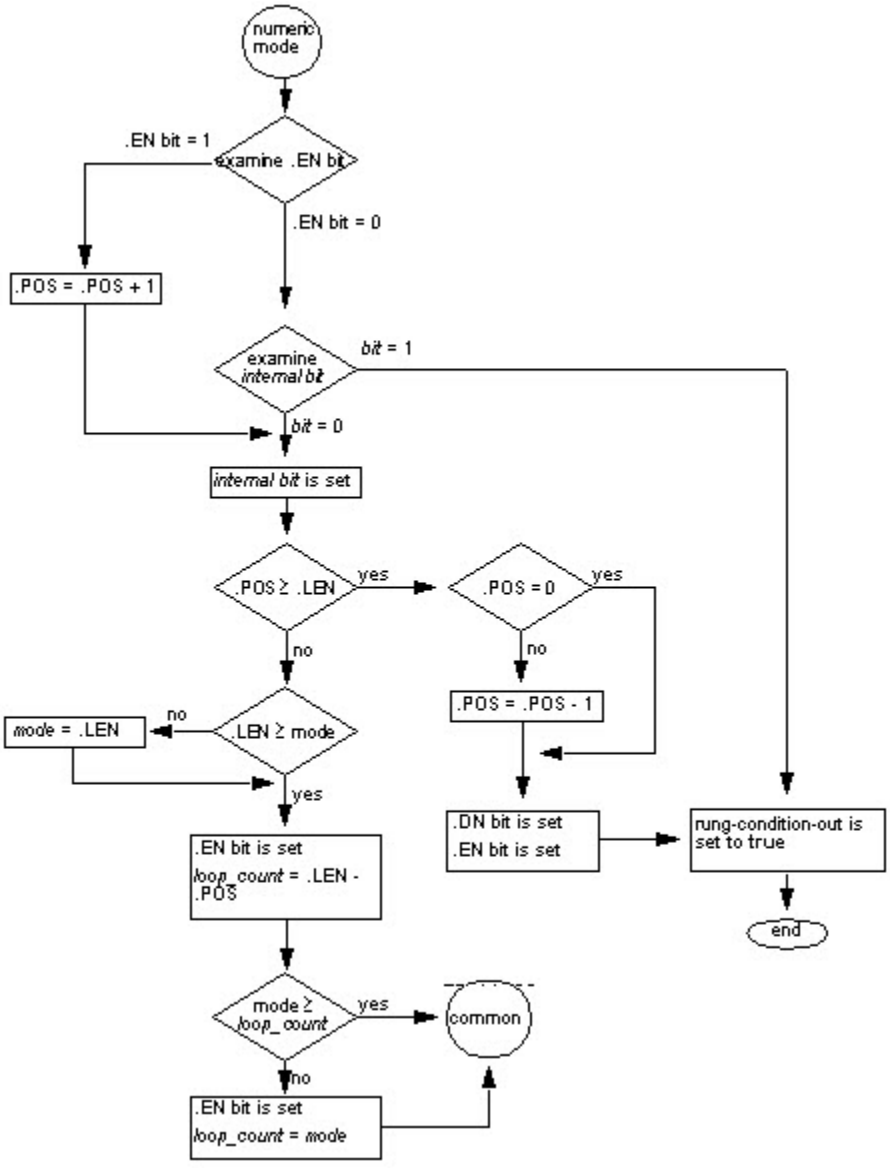
FAL 흐름도(렁-출력-조건이 참인 경우)



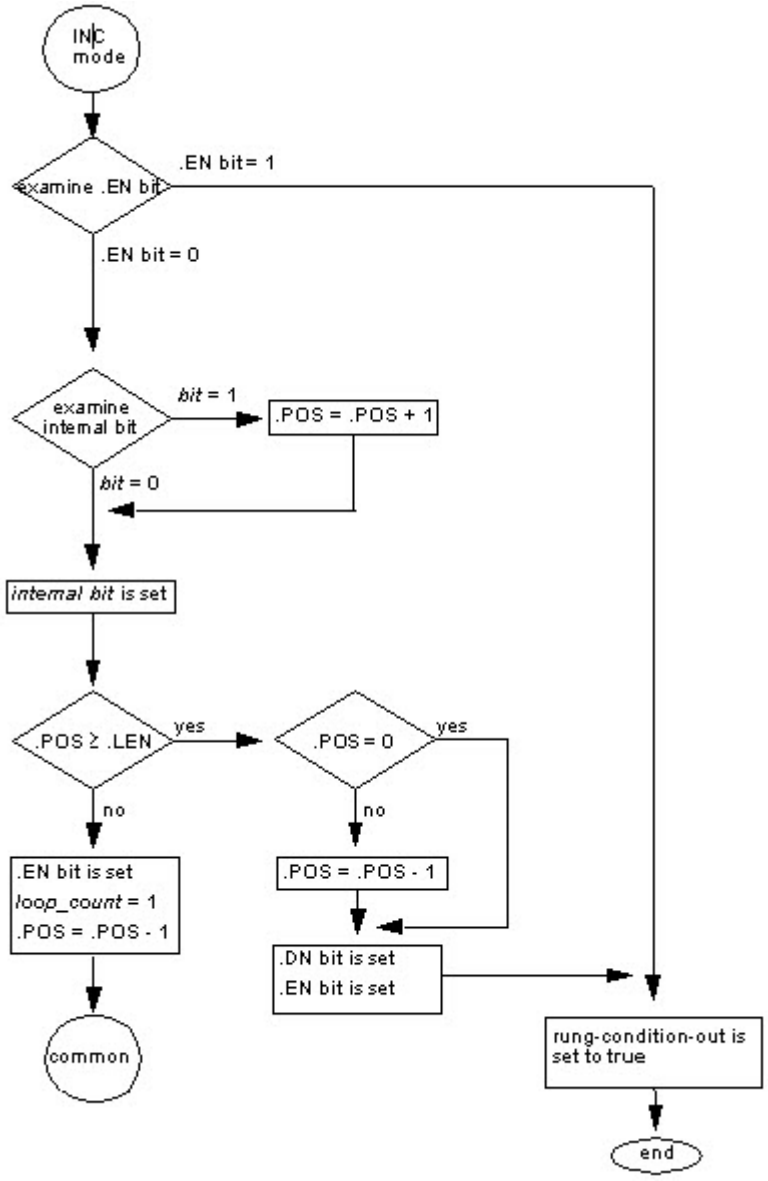
FAL 흐름도(전체 모드)


ALL mode
examine .EN bit
.EN bit = 0
examine internal bit
bit = 1
.EN bit = 1
bit = 0
.POS = .POS + 1
.EN bit is set
.POS ≥ .LEN
yes
.POS = 0
yes
no
no
loop_count = .LEN - .POS
.POS = .POS - 1
.POS = .POS - 1
common
.DN bit is set
.EN bit is set
rung-condition-out is set to true
end

FAL 흐름도(숫자 모드)


numeric mode
.EN bit = 1
examine .EN bit
.EN bit = 0
.POS = .POS + 1
examine internal bit
bit = 1
bit = 0
internal bit is set
.POS ≥ .LEN
yes
.POS = 0
yes
no
no
.POS = .POS - 1
mode = .LEN
no
.LEN ≥ mode
yes
.DN bit is set
.EN bit is set
rung-condition-out is set to true
end
.EN bit is set
loop_count = .LEN - .POS
mode ≥ loop_count
yes
common
no
.EN bit is set
loop_count = mode

FAL 흐름도(증가 모드)

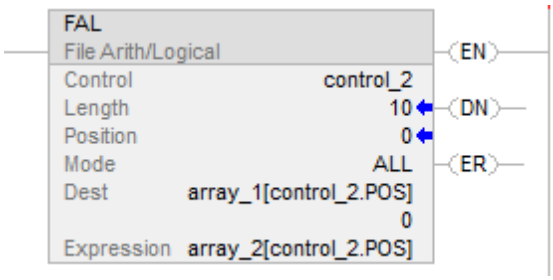


예제

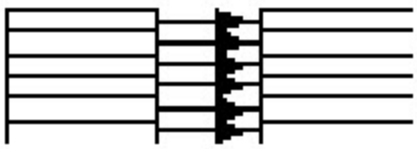
예 1

배열 -> 배열.

래더 다이어그램



활성화된 경우 FAL 명령어는 array_2 의 각 요소를 array_1 내의 동일한 위치에 복사합니다.



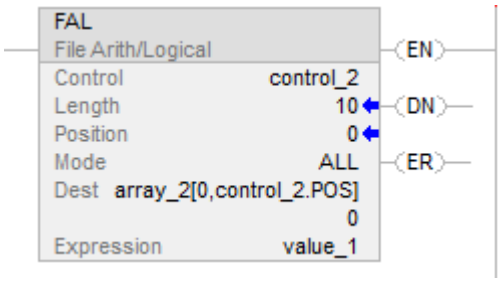
Expression:
array_2[control_2.pos]

Destination:
array_1[control_2.pos]

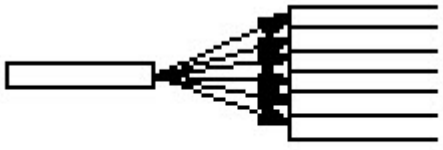
예 2

요소 -> 배열 복사.

래더 다이어그램



활성화된 경우 FAL 명령어는 value_1 을 array_2 의 두 번째 차원의 처음 10 개의 위치에 복사합니다.

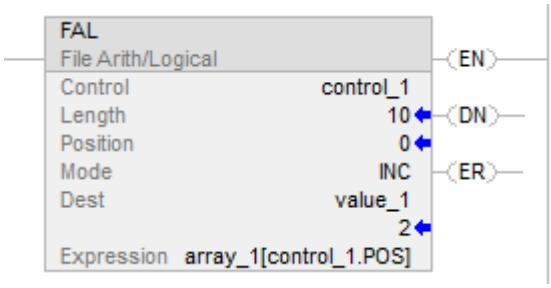


Expression:
value_1

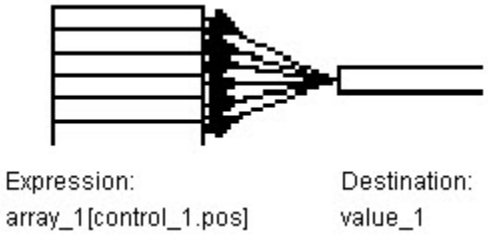
Destination:
array_2[0,control_2.pos]

예 3:

배열 -> 요소 복사.

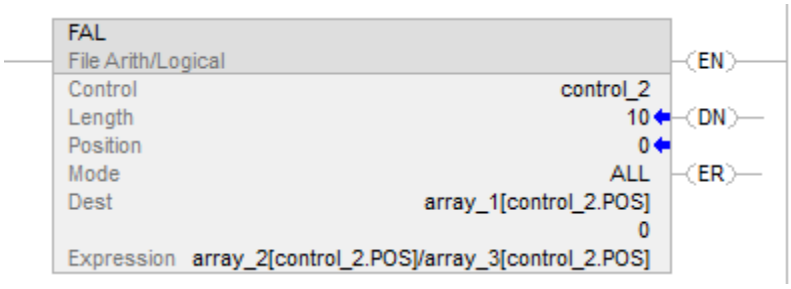


FAL 명령어는 활성화될 때마다 array_1 의 현재 값을 value_1 에 복사합니다. FAL 명령어는 증가 모드를 사용하므로 명령어가 활성화될 때마다 한 개의 배열 값만 복사됩니다. 다음에 명령어가 활성화되면 value_1 을 array_1 의 다음 값으로 덮어씁니다.

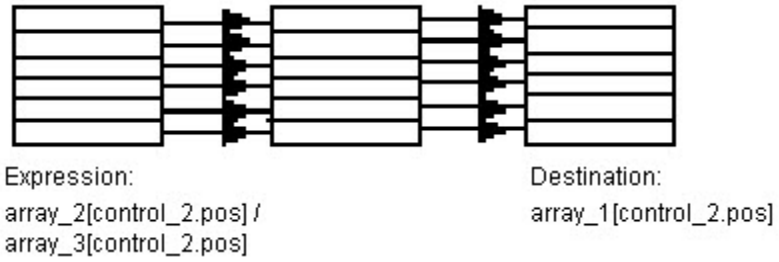


예 4:

산술 연산: 배열/배열 -> 배열

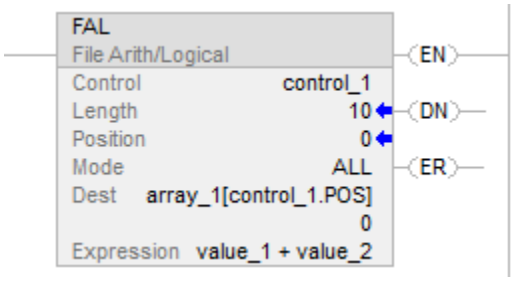


활성화된 경우 FAL 명령어는 array_2 의 현재 위치의 값을 array_3 의 현재 위치의 값으로 나누고 결과를 array_1 의 현재 위치에 저장합니다.

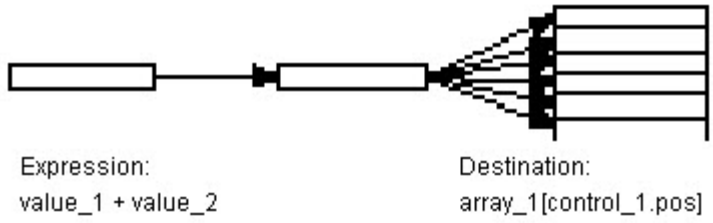


예 5:

산술 연산: 배열/배열 -> 배열

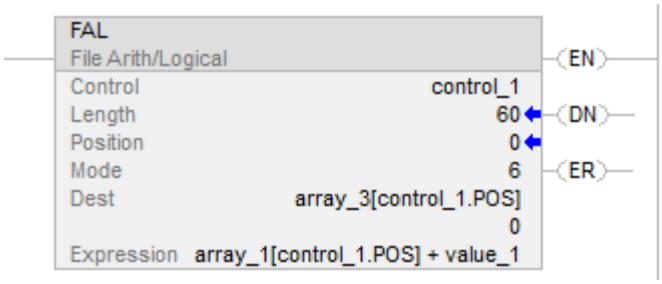


활성화된 경우 FAL 명령어는 value_1 과 value_2 를 더하고 결과를 array_1 의 현재 위치에 저장합니다.

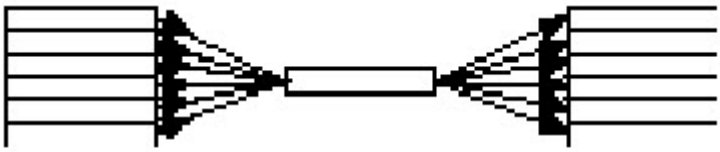


예 6:

산술 연산: 배열 + 요소 -> 배열



활성화된 경우 FAL 명령어는 array_1 의 현재 위치의 값을 value_1 에 더하고 결과를 array_3 의 현재 위치에 저장합니다. 명령어가 조작할 전체 array_1 및 array_3 에 대해 연산을 10 번 수행해야 합니다.

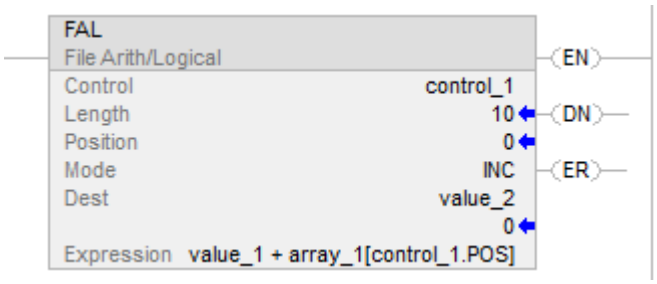


Expression:
array_1[control_1.pos] + value_1

Destination:
array_3[control_1.pos]

예 7:

산술 연산: (요소 + 배열) -> 요소



FAL 명령어가 활성화될 때마다 value_1 을 array_1 의 현재 값에 더하고 결과를 value_2 에 저장합니다. FAL 명령어는 증가 모드를 사용하므로 명령어가 활성화될 때마다 한 개의 배열 값만 value_1 에 더해집니다. 다음에 명령어가 활성화되면 value_2 를 덮어씁니다.

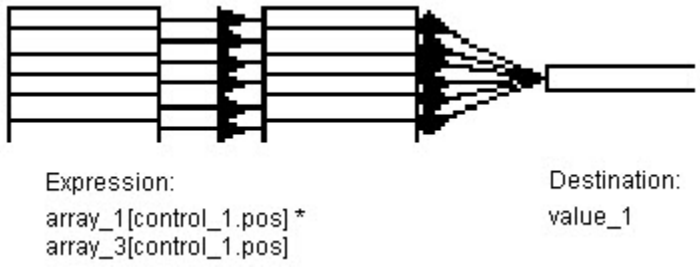
Expression:
value_1 + array_1[control_1.pos]

Destination:
value_2

예 8:

산술 연산: (배열 * 배열) -> 요소

활성화된 경우 FAL 명령어는 array_1 의 현재 값에 array_3 의 현재 값을 곱하고 결과를 value_1 에 저장합니다. FAL 명령어는 증가 모드를 사용하므로 명령어가 활성화될 때마다 한 쌍의 배열 값만 곱해집니다. 다음에 명령어가 활성화되면 value_1 을 덮어씁니다.



추가 참조

파일/기타 명령어 페이지의 545

유효한 연산자 페이지의 407

배열을 통한 인덱스 페이지의 978

데이터 변환 페이지의 967

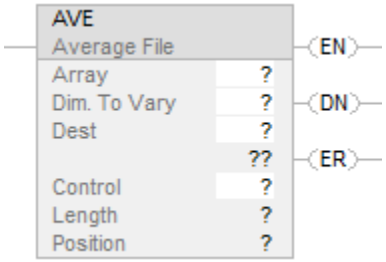
파일 평균(AVE)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

AVE 명령어는 한 세트의 값의 평균값을 계산합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
배열 태그	SINT INT DINT REAL	태그	이 배열 내 값의 평균값을 계산합니다 평균값 계산을 위한 요소 그룹의 첫 번째 요소를 지정합니다. 첨자에 CONTROL.POS 를 사용하지 마십시오.
Dimension to vary	DINT	즉시 (0, 1, 2)	사용할 차원 차원 순서: array[0,1,2]
Destination	SINT INT DINT REAL	태그	연산 결과
Control	CONTROL	태그	연산의 제어 구조
Length	DINT	즉시	평균값 계산에 사용할 배열 내 요소 수

Position	DINT	즉시	명령어가 액세스하고 있는 현재의 요소를 식별하는 지정 배열의 오프셋. 초기값은 일반적으로 0 입니다.

설명

AVE 명령어는 한 세트의 값의 평균값을 계산합니다.

중요: Length 로 인해 명령어가 지정된 Dimension to vary 를 초과하지 않게 해야 합니다. 초과하는 경우, 대상이 부정확해집니다. 자세한 내용은 배열을 메모리 블록으로 보기를 참조하십시오.

식 계산 중에 오버플로가 발생하는 경우, 명령어가 배열의 끝을 지나서 읽고 ER 비트를 설정하고 실행을 정지합니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, 연산 상태 플래그 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

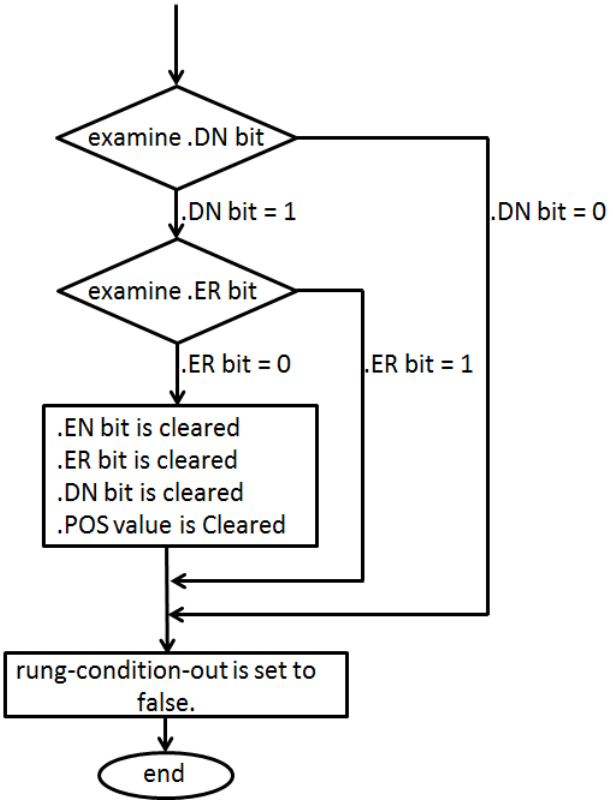
이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	.EN 비트가 해제됩니다. .DN 비트가 해제됩니다. 사전 스캔 중에 .ER 비트가 0 이면 모든 제어 비트(.DN, .EN, .EU, .EM, .UL, .IN, .FD)가 0 으로 해제됩니다.
렁-입력-조건이 거짓임.	AVE 흐름도(거짓)를 참조하십시오.
렁-입력-조건이 참임.	AVE 명령어는 배열 내 지정된 모든 요소를 더한 다음에 요소의 개수로 나누어서 평균값을 계산합니다.
사후 스캔	N/A.

AVE 흐름도(거짓)



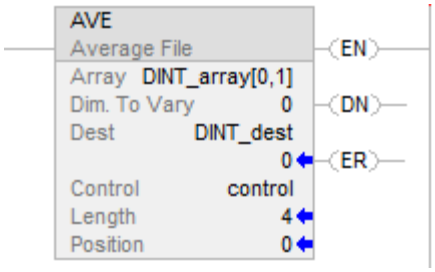
예 1

		dimension 1				
	subscripts	0	1	2	3	4
dimension 0	0	20	19	18	17	16
	1	15	14	13	12	11
	2	10	9	8	7	6
	3	5	4	3	2	1

$$AVE = \frac{19+14+9+4}{4} = \frac{46}{4} = 11.5$$

dint_ave = 12

래더 다이어그램



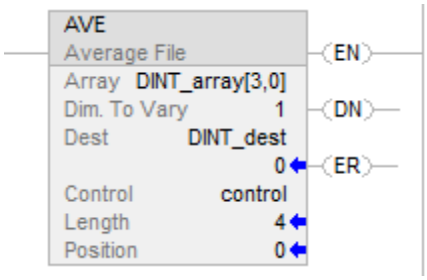
예 2

		dimension 1				
	subscripts	0	1	2	3	4
dimension 0	0	20	19	18	17	16
	1	15	14	13	12	11
	2	10	9	8	7	6
	3	5	4	3	2	1

$$AVE = \frac{5+4+3+2+1}{5} = \frac{15}{5} = 3$$

dint_ave = 3

래더 다이어그램



추가 참조

공통 속성 페이지의 963

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

파일 채우기(FLL)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

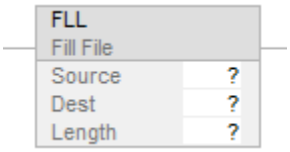
FLL 명령어가 메모리 블록을 제공된 소스 값으로 채웁니다. 소스는 변경되지 않습니다.

대상 배열이 SINT, INT, DINT 또는 REAL 이고 소스 값의 유형이 다른 경우 소스 값이 해당 대상 유형으로 변환된 후에 저장됩니다. 작은 정수 유형은 부호 확장을 통해 큰 정수로 변환됩니다.

대상 배열이 구조인 경우 소스 값이 변환 없이 기록됩니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	복사할 요소
Destination	SINT INT DINT REAL 구조	태그	Source 로 덮어쓸 초기 요소.
Length	DINT INT SINT	즉시 태그	채울 대상 요소의 수.

채워진 바이트 수가 다음 중 작은 값:

- 요청된 양 = Length x (대상 요소의 바이트 수)
- 대상 태그의 바이트 수

팁: 대상 태그의 끝이 기본 태그의 마지막 바이트로 정의됩니다. 태그가 구조인 경우, 태그의 끝은 구조의 마지막 요소의 마지막 바이트입니다. 이는 FLL 명령어가 구성원 배열의 끝을 지나서 작성할 수 있지만 어떠한 경우에도 기본 태그의 끝을 지나서 작성하지 않음을 의미합니다. FLL 명령어가 변경되어서는 안 되는 데이터를 변경하지 않는지 테스트하고 확인하십시오.

최상의 결과를 얻으려면 Source 와 Destination 의 유형이 동일해야 합니다. FLL 을 사용하여 구조를 상수(0 등)으로 채웁니다.

구조를 초기화할 경우 초기 값이 포함된 인스턴스가 하나여야 하며 COP 를 사용하여 이를 복제해야 합니다. 예를 들어 FLL 을 사용하여 전체 구조를 0 으로 해제할 수 있습니다.

Source 는 다음인 경우	Destination 은 다음인 경우	Source 다음 유형으로 변환
SINT, INT, DINT 또는 REAL	SINT	SINT
SINT, INT, DINT 또는 REAL	INT	INT
SINT, INT, DINT 또는 REAL	DINT	DINT
SINT, INT, DINT 또는 REAL	REAL	REAL

큰 정수에서 작은 정수로 변환되면 잘립니다(높은 비트가 삭제됨). 소스가 변환되면 대상에 N 번 기록되며, 여기에서 N 은 바이트 수입니다. 작은 정수에서 큰 정수로 변환되면 부호 확장이 발생합니다. 정수로 변환되면 REAL 값이 반올림됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

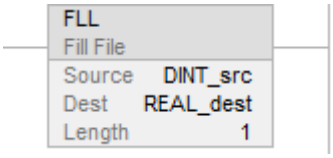
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 메모리를 채웁니다.
사후 스캔	N/A

예

FLL 명령어가 Length 에 지정된 여러 대상 요소를 DINT_src 유형 소스 피연산자에서 REAL_dest 유형 대상에 복사합니다.

래더 다이어그램



추가 참조

파일/기타 명령어 페이지의 545

배열을 통한 인덱스 페이지의 978

데이터 변환 페이지의 967

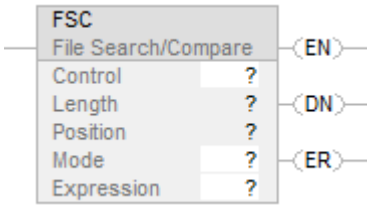
파일 검색 및 비교(FSC)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

FSC 명령어는 배열 내 값들을 요소마다 비교합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
제어	CONTROL	태그	연산의 제어 구조
길이(Length)	DINT	즉시	조작할 배열 내 요소의 수
위치(Position)	DINT	즉시	배열로 오프셋 초기값은 일반적으로 0 입니다.

모드(Mode)	DINT	즉시	작업을 분배하는 방식 INC, ALL 을 선택하거나, 1 ~ 2147483647 범위의 수를 입력함
식(Expression)	SINT INT DINT REAL STRING	즉시 태그	태그 및/또는 연산자로 구분된 즉시 값으로 구성된 식

CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 FSC 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 마지막 요소에서 작업한 경우에 설정됩니다(.POS = .LEN).
.ER	BOOL	에러 비트는 수정되지 않습니다.
.IN	BOOL	금지 비트는 FSC 명령어가 참인 비교를 감지했다는 것을 나타냅니다. 검색 작업을 계속하려면 이 비트를 지워야 합니다.
.FD	BOOL	발견 비트는 FSC 명령어가 참인 비교를 감지했다는 것을 나타냅니다.
.LEN	DINT	길이는 명령어가 작업할 배열 내 요소의 수를 지정합니다.
.POS	DINT	위치에는 명령어가 현재 액세스하고 있는 요소의 위치가 표시됩니다.

설명

FSC 명령어의 EnableIn 이 거짓에서 참으로 전환되는 경우, 식의 값이 지정된 반복 모드 동안 계산됩니다.

평가 결과가 참인 경우, 명령어는 .FD 비트를 설정하며 .POS 값은 명령어가 참인 비교를 발견한 배열 위치를 반영합니다. 명령어는 .IN 비트를 설정하여 추가 반복 작업을 방지합니다.

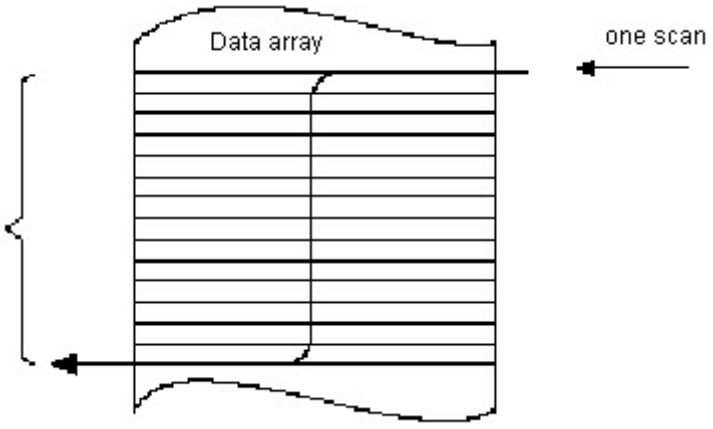
연산 모드 선택하기

FSC 명령어의 경우, 모드는 컨트롤러에 배열 작업을 분배하는 방법을 명령합니다.

실행할 작업:	선택할 모드:
다음 명령어로 계속 실행하기 전에 지정된 배열 내 모든 요소에서 작업합니다.	모두
배열 작업을 여러 스캔에 분배합니다. 스캔당 작업할 요소의 수(1 ~ 2147483647)를 입력합니다.	숫자
EnableIn 이 거짓에서 참으로 바뀔 때마다 배열 내 요소 하나를 조작합니다.	증가

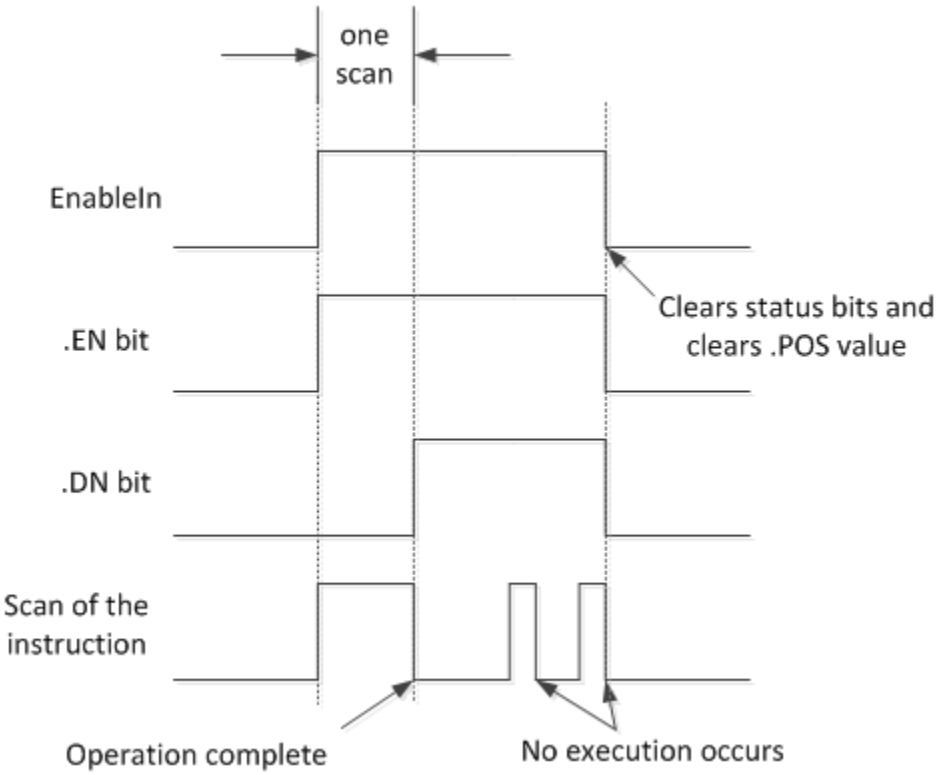
전체 모드

전체 모드에서는 다음 명령어로 계속 실행하기 전에 배열 내 모든 지정된 요소에서 작업합니다. 작업은 명령어의 EnableIn 이 거짓에서 참으로 바뀔 때 시작합니다. 제어 구조의 위치(.POS) 값은 명령어가 현재 사용하고 있는 배열 내 요소를 가리킵니다. 작업이 다음과 같은 두 가지 경우에 정지합니다. .POS 값이 .LEN 값과 같거나 이를 초과하는 경우 논리곱 식이 참으로 계산되는 경우입니다.



아래 타이밍 다이어그램에서는 상태 비트와 명령어 작업의 관계를 보여줍니다. 명령어 실행이 완료되면 .DN 비트가 참입니다.

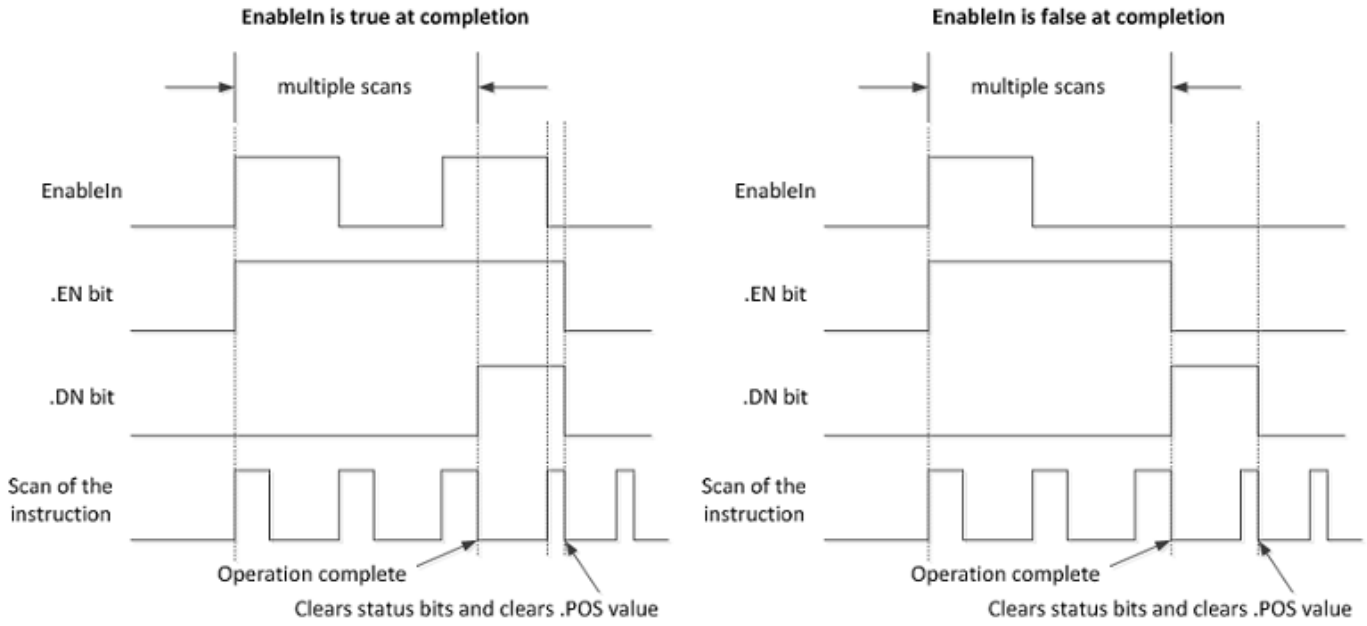
EnableIn 이 거짓이면 .DN 비트, .EN 비트, .POS 값이 지워집니다. 이 경우에만 EnableIn 거짓 - 참 전환에서 명령어의 다른 실행이 트리거될 수 있습니다.



숫자 모드

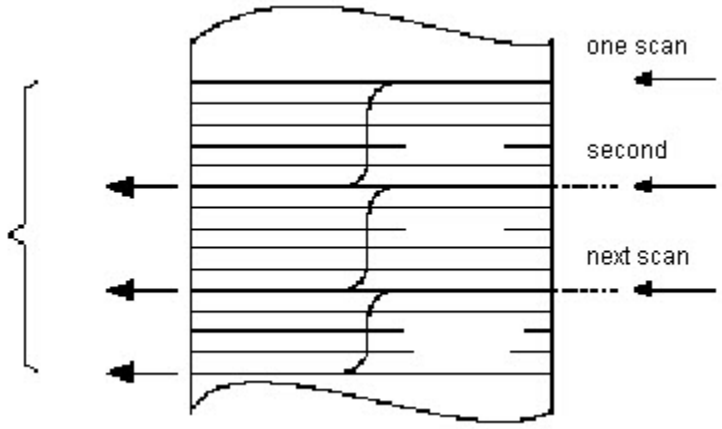
숫자 모드는 배열 작업을 여러 스캔 작업에 분배합니다. 이 모드는 비시간 임계 데이터나 대량의 데이터를 작업할 경우에 유용합니다. 스캔마다 작업할 요소의 수를 입력하여 스캔 시간을 줄일 수 있습니다.

EnableIn 이 거짓에서 참으로 전환되면 실행이 트리거됩니다. 이 경우 명령어는 전체 배열의 작업을 완료하는 데 필요한 스캔 작업의 횟수 만큼 스캔될 때마다 실행됩니다. 트리거되면 EnableIn 은 명령어의 실행을 중단하지 않고도 반복해서 변경할 수 있습니다.



.DN 또는 .IN 비트가 참이 될 때까지는 숫자 모드로 작업하는 파일 명령어의 결과 사용을 피합니다.

아래 타이밍 다이어그램에서는 상태 비트와 명령어 작업의 관계를 보여줍니다. 명령어 실행이 완료되면 .DN 비트가 참입니다.

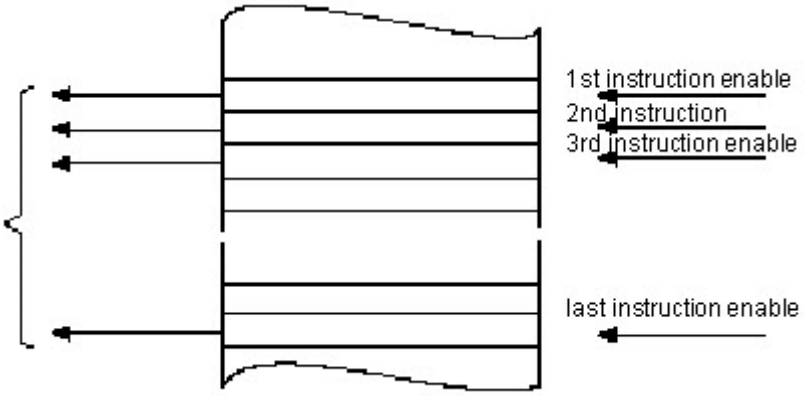


완료 시에 EnableIn 이 참이면 .EN 과 .DN 비트는 EnableIn 이 거짓으로 바뀔 때까지 참입니다. EnableIn 이 거짓으로 바뀌면 이들 비트와 .POS 값은 지워집니다.

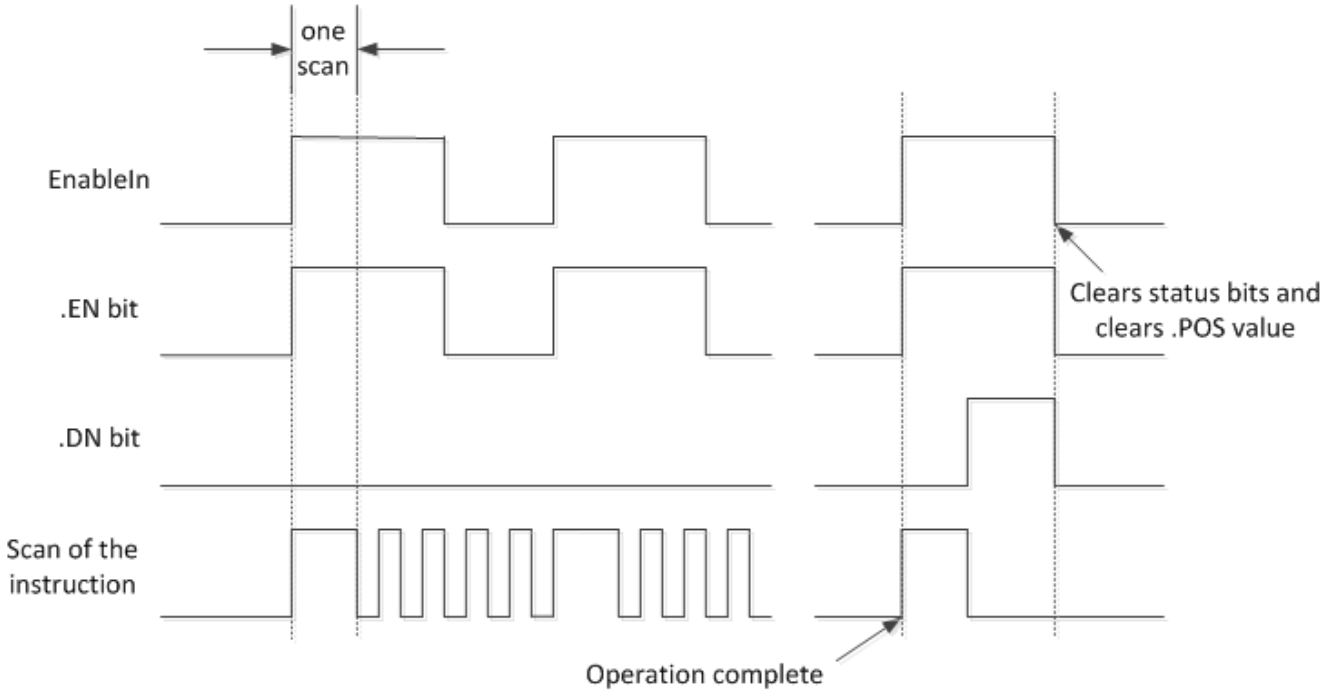
완료 시에 EnableIn 이 거짓이면 .EN 비트는 즉시 지워집니다. .DN 비트와 .POS 값은 .EN 비트가 지워진 직후의 스캔 작업 시에 지워집니다.

증가 모드

증가 모드는 명령어의 EnableIn 이 거짓에서 참으로 바뀔 때마다 배열 내 요소 하나를 조작합니다.



아래 타이밍 다이어그램에서는 상태 비트와 명령어 작업의 관계를 보여줍니다. 실행은 EnableIn 이 거짓에서 참으로 바뀌는 스캔에서만 진행됩니다. 이 상황이 발생할 때마다 배열 내 요소 하나만 조작됩니다. EnableIn 이 둘 이상의 스캔에서 참으로 유지하는 경우, 명령어는 첫 번째 스캔에만 실행됩니다.



렁-입력-조건이 참인 경우, .EN 비트가 설정됩니다. 배열 내 마지막 요소가 조작되면 .DN 비트가 설정됩니다. 마지막 요소가 조작되고 렁-입력-조건이 거짓으로 전환되면 .EN 비트, .DN 비트, .POS 값이 지워집니다.

스캔당 요소 하나의 속도에 따른 증가 모드와 숫자 모드의 차이점은 다음과 같습니다.

스캔당 요소가 몇 개 있는 숫자 모드의 실행을 시작하려면 EnableIn 의 거짓 - 참 전환이 하나 있으면 됩니다. 명령어는 EnableIn 의 상태에 상관 없이 완료될 때까지 스캔마다 지정된 개수의 요소를 계속 실행합니다.

증가 모드는 EnableIn 이 거짓에서 참으로 전환되어야 배열 내 요소 하나를 작업할 수 있습니다.

포맷 식

식에서 사용하는 각 연산자에는 1 ~ 2 개의 피연산자(태그 또는 즉시 값)를 제공해야 합니다. 다음 표를 사용하여 식 내의 연산자 및 피연산자 형식을 지정합니다.

연산자 작용 대상:	사용 형식:	예
피연산자 1 개	연산자(피연산자)	ABS(tag)
피연산자 2 개	operand_a 연산자 operand_b	tag_b + 5 tag_c AND tag_d (tag_e**2) MOD (tag_f / tag_g)

연산 순서 정의

명령어는 식 안에 써넣은 연산을 작성 순서가 아닌 미리 규정된 순서에 따라 수행합니다. 작업 순서를 오버라이드하려면 해당 항을 모아 괄호 안에 넣음으로써 명령어가 다른 연선보다 우선 괄호 안의 연선을 수행하게 하면 됩니다.

연산 순서가 같은 경우에는 왼쪽에서 오른쪽 순으로 수행됩니다.

순서	연산
1	( )
2	ABS, ACS, ASN, ATN, COS, DEG, FRD, LN, LOG, RAD, SIN, SQR, TAN, TOD, TRN
3	**
4	- (부정), NOT
5	*, /, MOD
6	- (빼기), +
7	논리곱
8	XOR
9	또는
10	<, <=, >, >=, =, <>

식에 문자열 사용하기

식에 ASCII 문자로 된 문자열을 사용하려면 다음 지침을 따르십시오.

식을 통해 두 문자열 태그를 비교할 수 있습니다.

ASCII 문자는 식에 직접 입력할 수 없습니다.

아래와 같은 연산자만 허용됩니다.

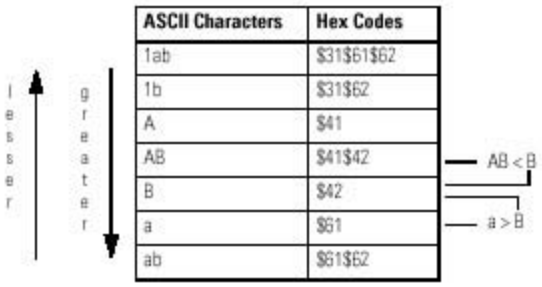
연산자	설명
=	같음
<	보다 작음
<=	보다 작거나 같음
>	보다 큼
>=	보다 크거나 같음
<>	같지 않음

문자들이 일치하면 문자열이 같습니다.

ASCII 문자는 대소문자가 구분됩니다. 대문자 A($41)와 소문자 a($61)는 다릅니다.

16 진수 문자값으로 한 문자열이 다른 문자열보다 작거나 큰지를 판단합니다.

두 문자열이 전화번호부와 마찬가지로 정렬된 경우 문자열의 순서로 어느 문자열이 큰지 알 수 있습니다.



연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
ControlLogix 5580	아니요
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570	예

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
.POS < 0 또는 .LEN < 0	4	21

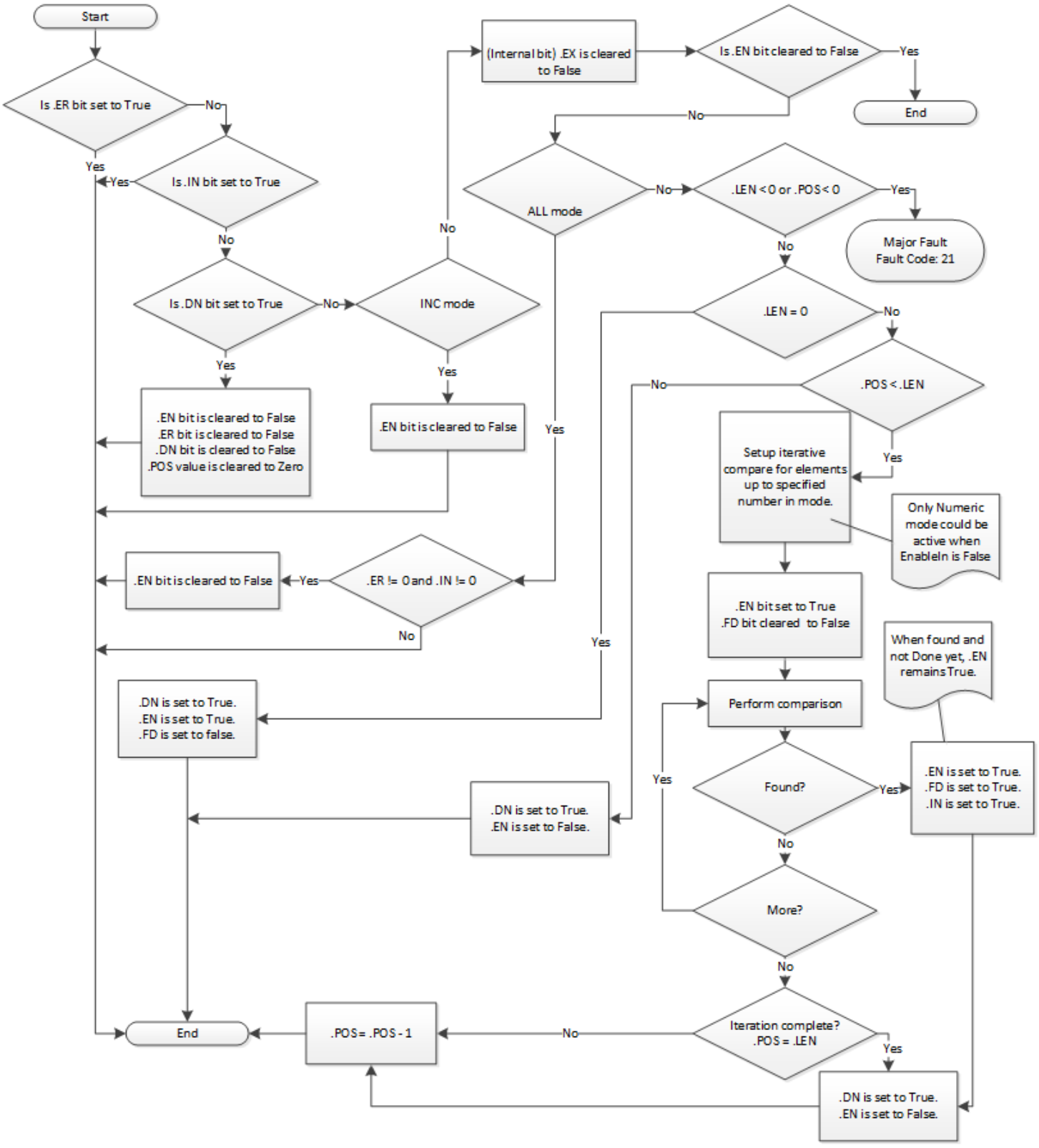
피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오. 배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

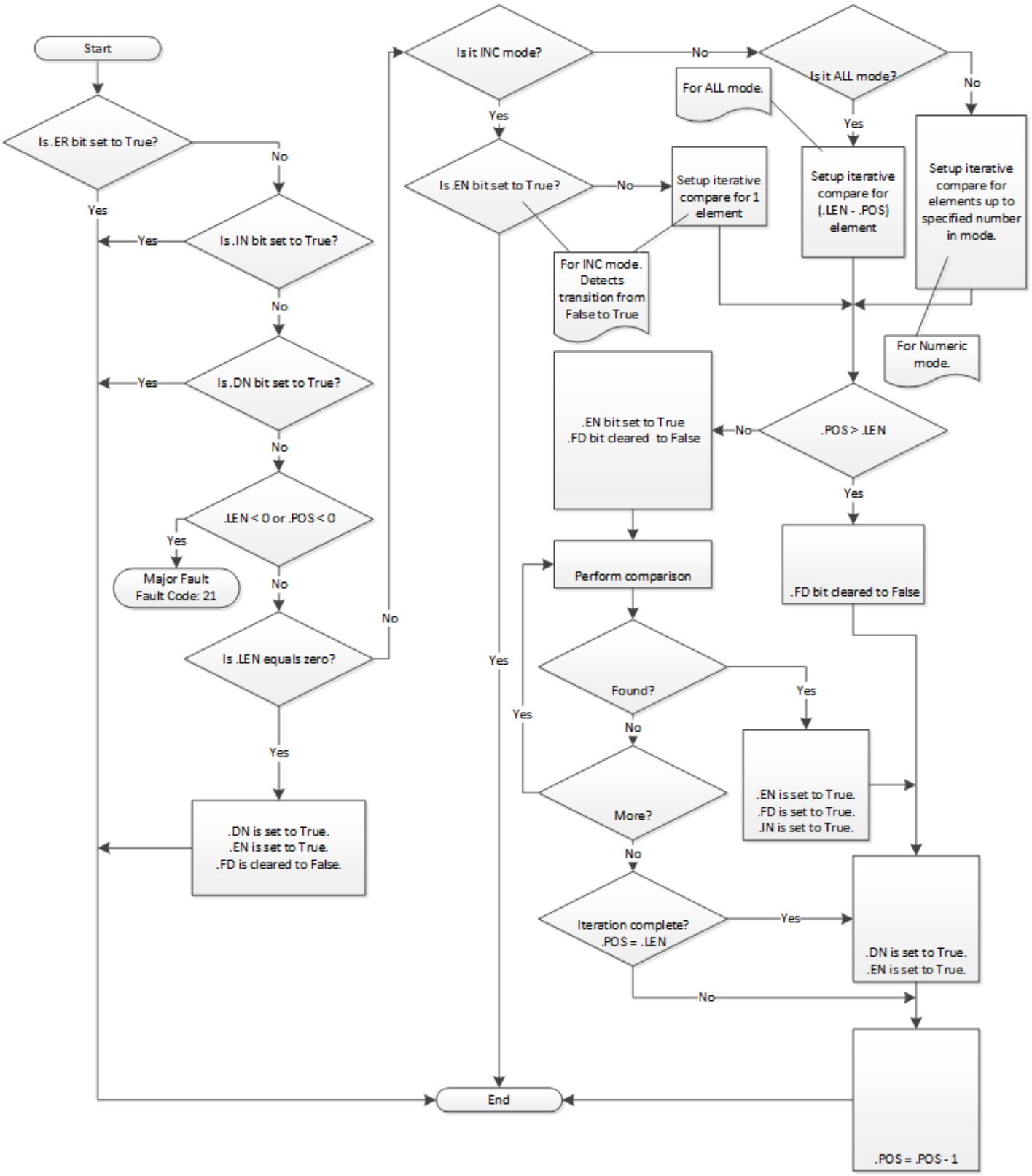
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	FSC 흐름도(렁-출력-조건이 거짓임)를 참조하십시오.
렁-입력-조건이 참임	FSC 흐름도(렁-출력-조건이 참임)를 참조하십시오.
사후 스캔	N/A

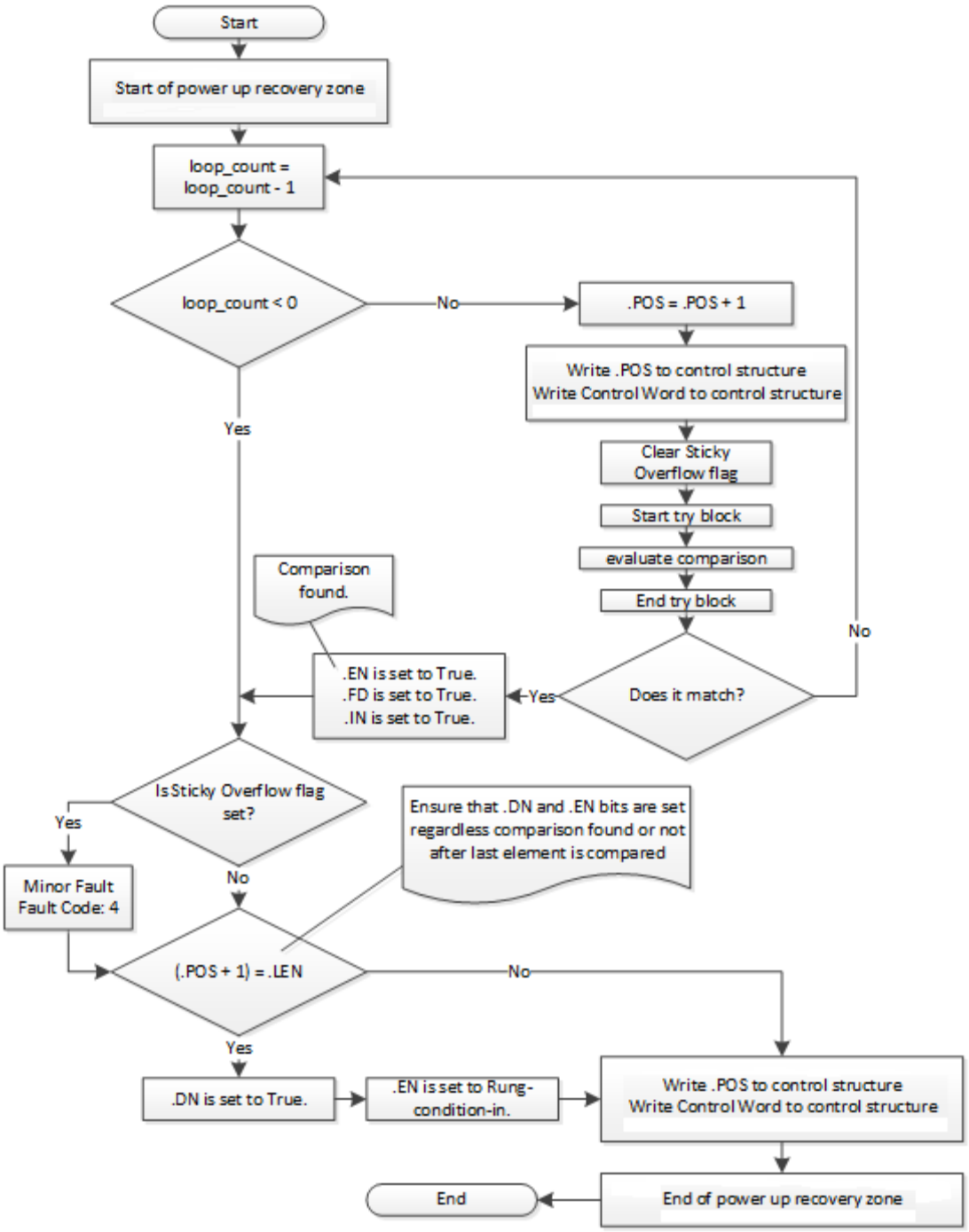
FSC 흐름도(렁-출력-조건이 거짓임)



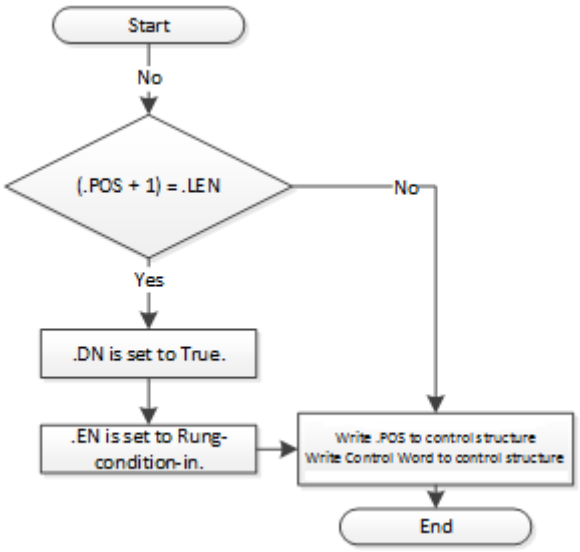
FSC 흐름도(렁-출력-조건이 참임)



FSC 흐름도(FSC 보통 서브플로)



FSC 흐름도(FSC 보통 예외 서브플로)

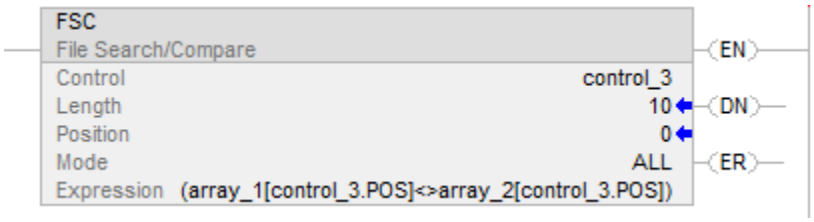


예제

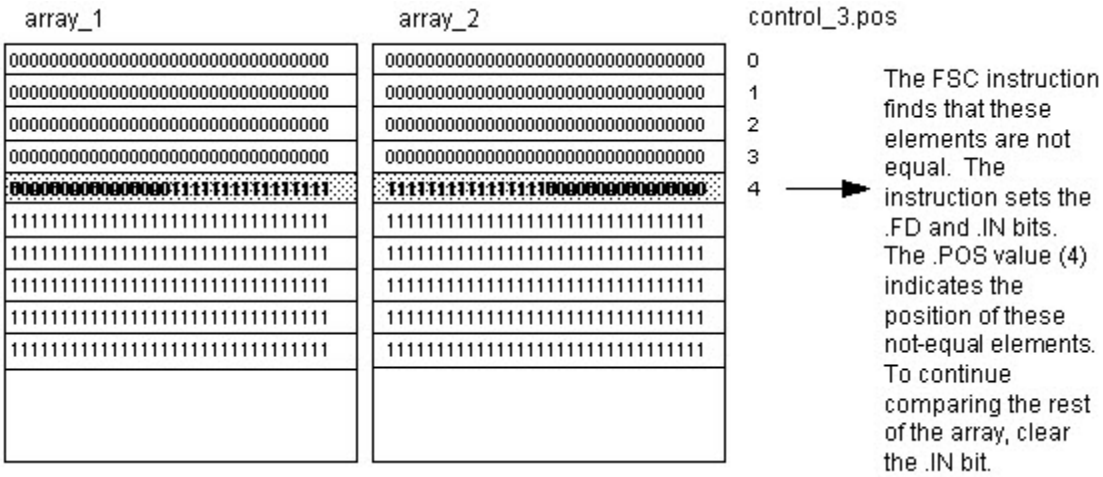
예 1

두 DINT 배열에서 동일하지 않은 요소 검색합니다.

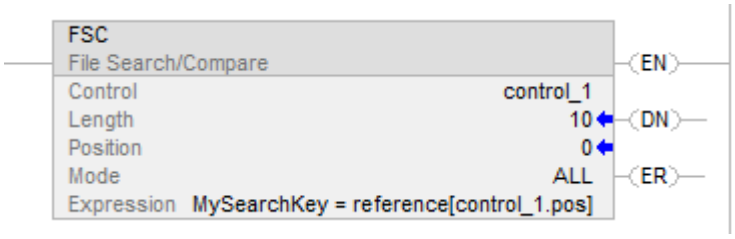
래더 다이어그램



활성화된 경우 FSC 명령어는 array_1 의 처음 10 개의 요소를 각각 array_2 의 해당 요소와 비교합니다. 동일하지 않은 요소가 발견된 경우, FD 및 IN 비트가 설정됩니다. POS 는 동일하지 않은 요소의 위치를 식별합니다. 배열의 나머지 부분을 검색하려면 IN 비트를 지웁니다.

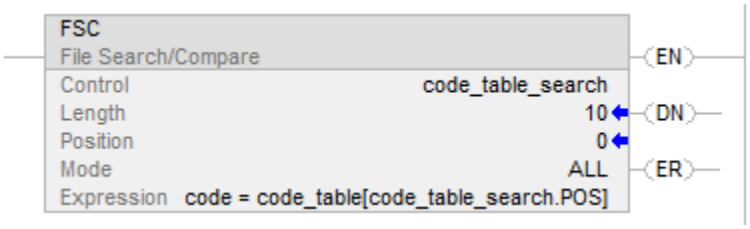


예 2



구조 배열에서 일치하는 부분을 검색합니다.

예 3



문자열 배열에서 특정 문자열을 검색합니다.

활성화된 경우, FSC 명령어는 코드의 문자와 code_table 의 10 개 요소를 비교합니다.

추가 참조

파일/기타 명령어 페이지의 545

CMP 페이지의 330

FAL 페이지의 556

배열을 통한 인덱스 페이지의 978

유효한 연산자 페이지의 407

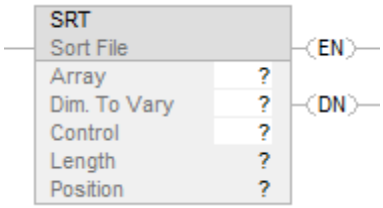
파일 정렬(SRT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

SRT 명령어가 값 세트를 배열의 한 차원(Dim to vary)에 오름차순으로 정렬합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

SRT(Array,Dimtovary,Control);

피연산자

래더 다이어그램

피연산자	유형	형식	설명
배열	SINT INT DINT REAL	배열 태그	정렬할 배열 정렬할 요소 그룹의 첫 번째 요소 지정
변할 차원	DINT	즉시 (0, 1, 2)	사용할 차원 차원 순서: array[0,1,2]
제어(Control)	CONTROL	태그	연산의 제어 구조
길이(Length)	DINT	즉시	정렬할 배열의 요소 수
위치(Position)	DINT	즉시	배열의 현재 요소 초기값은 일반적으로 0 입니다.

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
배열	SINT INT DINT REAL	배열 태그	정렬할 배열 정렬할 요소 그룹의 첫 번째 요소 지정
변할 차원(Dimension to vary)	DINT	즉시 (0, 1, 2)	사용할 차원 차원 순서: array[0,1,2]
제어(Control)	CONTROL	태그	연산의 제어 구조
길이(Length)	DINT	즉시	정렬할 배열의 요소 수. 지정된 Length 및 Position 값은 CONTROL 구조의 .LEN 및 .POS 구성원에서 액세스합니다.
위치(Position)	DINT	즉시	배열의 현재 요소 초기값은 일반적으로 0 입니다. 지정된 Length 및 Position 값은 CONTROL 구조의 .LEN 및 .POS 구성원에서 액세스합니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 ST(스트럭처드 텍스트) 구문을 참조하십시오.

CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 SRT 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 배열의 마지막 요소에서 작동된 경우에 설정됩니다.
.ER	BOOL	에러 비트는 .LEN < 0 또는 .POS < 0 인 경우에 설정됩니다. 이러한 조건 중 하나가 충족되어도 메이저 폴트가 발생합니다. .ER 비트가 설정된 경우 명령어가 실행되지 않습니다.
.LEN	DINT	길이 워드는 명령어가 작동되는 배열의 요소 수를 지정합니다.
.POS	DINT	위치 워드는 명령어가 액세스하는 현재 요소를 식별합니다.

설명

SRT 명령어가 값 세트를 배열의 한 차원(Dim to vary)에 오름차순으로 정렬합니다.

중요: 명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

SRT 명령어는 연속 데이터 메모리에서 작동됩니다. CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러의 경우에만 명령어의 범위가 기본 태그로 제한됩니다. SRT 명령어가 기본 태그의 외부에 데이터를 쓰지 않지만 구성원 경계를 넘어설 수 있습니다. 구조의 구성원인 배열을 지정하고 길이가 해당 배열의 크기를 초과할 경우 SRT 명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에서는 데이터가 지정된 구성원으로 제한됩니다.

이 전환 명령어에서 명령어가 실행되려면 릴레이 래더가 렁-입력-조건을 거짓에서 참으로 토글 전환합니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	아니요
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
.POS < 0 또는 .LEN < 0	4	21
Dimension to vary > 차원 수	4	20
Length > 배열의 끝	4	20

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A.
렁-입력-조건이 거짓임	.EN 비트가 거짓으로 해제되었음 .EN 비트가 거짓으로 해제되었음 .DN 비트가 거짓으로 해제되었음
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A.

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	이 명령어를 실행하려면 전환이 필요하므로 거짓으로 실행된 다음 참으로 실행됩니다. 자세한 내용은 래더 다이어그램 표를 참조하십시오.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 참조하십시오.

예제

예 1

DINT[4,5]에 해당하는 DINT_array 를 정렬합니다.

Before

dimension 0 \ dimension 1	0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	10	9	8	7	6
3	5	4	3	2	1

After

dimension 0 \ dimension 1	0	1	2	3	4
0	20	19	3	17	16
1	15	14	8	12	11
2	10	9	13	7	6
3	5	4	18	2	1

래더 다이어그램

ST(스트럭처드 텍스트)

```
IF sort1 then
  control_1.LEN := 4;
  control_1.POS := 0;
  SRT(DINT_array[0,2],0, control_1);
END_IF;
```

예 2

DINT[4,5]에 해당하는 DINT_array 를 정렬합니다.

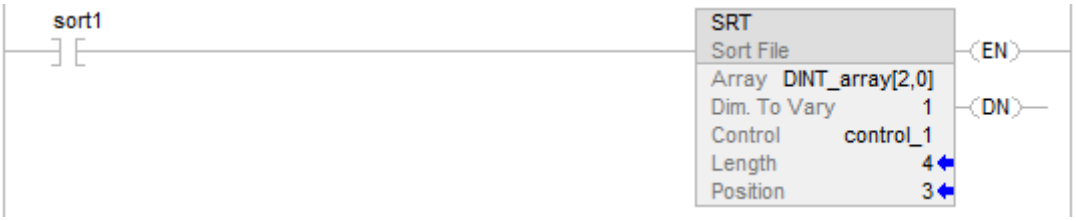
Before

dimension 0 \ dimension 1	0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	10	9	8	7	6
3	5	4	3	2	1

After

dimension 0 \ dimension 1	0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	6	7	8	9	10
3	5	4	3	2	1

래더 다이어그램



ST(스트럭쳐드 텍스트)

```
ctrl.LEN := 4;

ctrl.POS := 0;

SRT(DINT_array[0,2],0, ctrl);
```

추가 참조

파일/기타 명령어 페이지의 545

파일 평균(AVE) 페이지의 573

데이터 변환 페이지의 967

공통 특성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

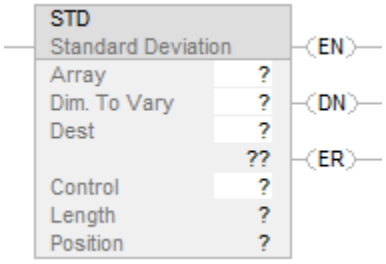
파일 표준 편차(STD)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

STD 명령어는 Array 의 한 차원에 있는 값의 표준 편차를 계산하고 그 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
배열	SINT INT DINT REAL	배열 태그	이 배열에 있는 값의 표준 편차를 찾습니다. 표준 편차를 계산하는 데 사용할 요소 그룹의 첫 번째 요소를 지정합니다.
변할 차원	DINT	즉시 (0, 1, 2)	사용할 차원 차원 순서: array[0,1,2]
Destination	REAL	태그	연산 결과

제어	CONTROL	태그	연산의 제어 구조
길이(Length)	DINT	즉시	표준 편차를 계산하는 데 사용할 배열의 요소 수
위치(Position)	DINT	즉시	명령어가 액세스하고 있는 현재의 요소를 식별하는 지정 배열의 오프셋. 초기값은 일반적으로 0 입니다.

CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 STD 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 배열의 마지막 요소에서 작동된 경우에 설정됩니다.
.ER	BOOL	에러 비트는 명령어가 오버플로를 일으킬 경우에 설정됩니다. 프로그램이 .ER 비트를 지울 때까지 명령어가 실행을 중지합니다. .POS 값은 오버플로를 유발한 요소의 위치를 저장합니다.
.LEN	DINT	길이 워드는 명령어가 작동되는 배열의 요소 수를 지정합니다.
.POS	DINT	위치 워드는 명령어가 액세스하고 있는 현재 요소를 식별하는 지정 배열의 오프셋입니다.

설명

표준 편차는 다음 공식에 따라 계산됩니다.

$$\text{Standard Deviation} = \sqrt{\frac{\left(\sum_{i=1}^{N}\left[\langle X_{(start+i)} - AVE\rangle^2\right]\right)}{(N-1)}}$$

여기에서:

start = 배열 피연산자의 dimension-to-vary 첨자

xi = 배열의 변수 요소

N = 배열의 지정된 요소 수

$$AVE = \frac{\left(\sum_{i=1}^{N'} x_{(start+i)} \right)}{N}$$

중요:	Length 로 인해 명령어가 지정된 Dimension to vary 를 초과하지 않게 해야 합니다. 초과하는 경우, Destination 이 부정확해집니다.

식 계산 중에 오버플로가 발생하거나 명령어가 배열의 끝을 지나서 읽기 작업을 수행할 경우 ER 비트를 설정하고 실행을 정지합니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	프로그래밍 언어에 따라 조건부. 연산 상태 플래그를 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
.POS < 0 또는 .LEN < 0	4	21
Dimension to vary > 차원 수	4	20

피연산자 관련 폴트에 대해서는 공통 특성을 참조하십시오.

실행

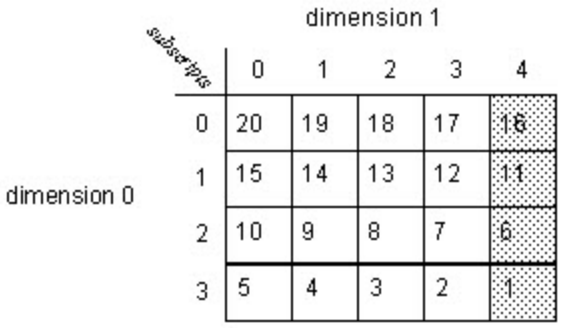
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	.EN 비트가 해제됩니다. .DN 비트가 해제됩니다. .ER 비트가 해제됩니다.
렁-입력-조건이 거짓임	.EN 비트가 해제됩니다. .ER 비트가 해제됩니다. .DN 비트가 해제됩니다. .POS 값이 해제됩니다. 렁-출력-조건이 거짓입니다.
렁-입력-조건이 참임	내부적으로 명령어가 FAL 명령어를 사용하여 평균값을 계산합니다. 식 = 표준 편차 계산 모드 = 전체
사후 스캔	N/A.

예제

예 1

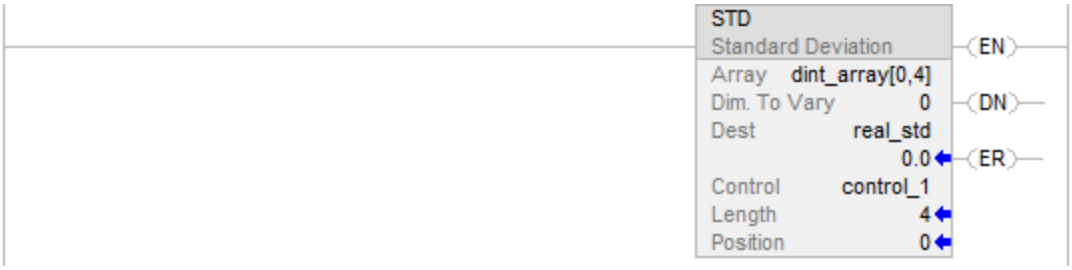
DINT[4,5]인 arrayDint 의 표준 편차를 계산합니다.



$$STD = \sqrt{\frac{\langle 16-8.5\rangle^2 + \langle 11-8.5\rangle^2 + \langle 6-8.5\rangle^2 + \langle 1-8.5\rangle^2}{\langle 4-1\rangle}} = 6.454972$$

$real\_std = 6.454972$

래더 다이어그램



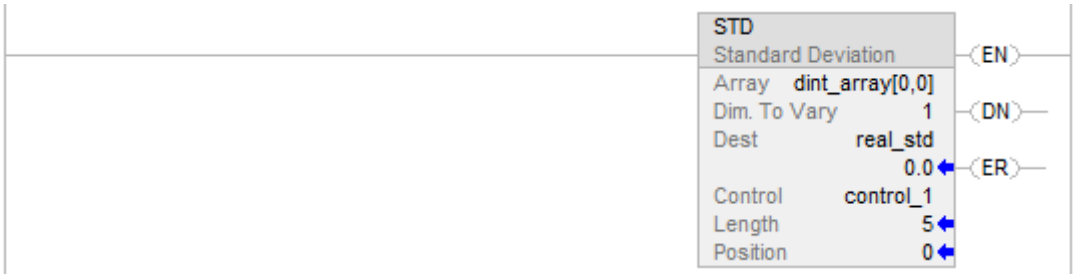
예 2

DINT[4,5]인 dint_array 의 표준 편차를 계산합니다.

dimension 1

dimension 0	0	1	2	3	4
0	20	19	18	17	16
1	15	14	13	12	11
2	10	9	8	7	6
3	5	4	3	2	1

래더 다이어그램



추가 참조

파일/기타 명령어 페이지의 545

AVE 페이지의 573

공통 특성 페이지의 963

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

요소의 크기(SIZE)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

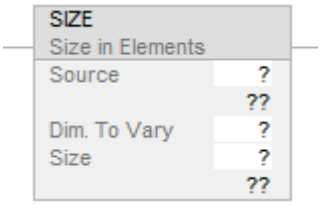
SIZE 명령어가 Source 배열의 지정된 차원 또는 문자열 피연산자에서 요소(크기) 수를 찾고 결과를 Size 피연산자에 놓습니다. 명령어가 배열의 한 차원의 크기를 찾습니다.

명령어가 다음에서 작동됩니다.

- 배열
- 구조의 배열
- 큰 배열에 속한 배열
- 문자열 태그

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

SIZE(Source,Dimtovary,Size);

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

명령어 내에서 숫자 데이터 유형을 혼합하기 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형	형식	설명	
Source	SINT INT DINT REAL 구조 문자열 유형	배열 태그	명령어가 작동되는 배열의 첫 번째 요소 검증 도중 수락되지 않은 배열이 아닌 태그	
Dimension to Vary	DINT	즉시 (0, 1, 2)	사용할 차원:	
			다음 차원의 크기:	**입력:**
			첫 번째 차원	0
			두 번째 차원	1
			세 번째 차원	2
Size	SINT INT DINT REAL	태그	배열의 지정된 차원에 요소 수를 저장하는 태그	

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 *배열을 통한 인덱스*를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	렁-출력-조건을 렁-입력-조건으로 설정합니다.
렁-입력-조건이 참임	렁-출력-조건을 렁-입력-조건으로 설정합니다. 명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

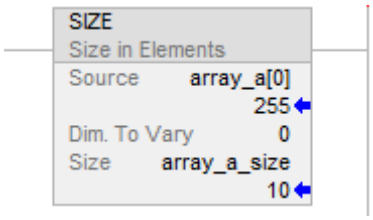
조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 확인하십시오.

예

예 1

array_a 의 차원 0(첫 번째 차원)에서 요소 수를 찾습니다. 크기를 array_a_size 에 저장합니다. 이 예에서는 array_a 의 차원 0 에 10 개의 요소가 있습니다.

래더 다이어그램



ST(스트럭처드 텍스트)

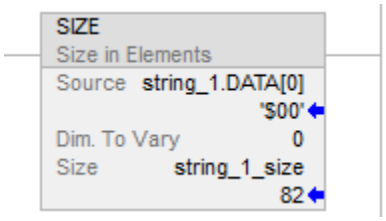
```
SIZE(array_a,0,array_a_size);
```

예 2

문자열인 string_1 의 DATA 구성원에서 요소 수를 찾습니다. 크기를 string_1_size 에 저장합니다.

이 예에서는 string_1 의 DATA 구성원에 82 개의 요소가 있습니다. 문자열이 기본 STRING 데이터 유형을 사용합니다. 각 요소에 한 개의 문자가 있으므로 string_1 에 최대 82 개의 문자가 포함될 수 있습니다.

래더 다이어그램



ST(스트럭처드 텍스트)

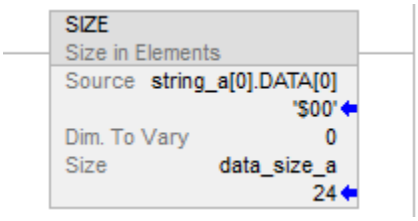
```
SIZE(string_1.DATA[0],0,string_1_size);
```

예 3

String_a 는 문자열 구조의 배열입니다. SIZE 명령어가 문자열 구조의 DATA 구성원에서 요소 수를 찾고 크기를 data_size_a 에 저장합니다.

이 예에서는 DATA 구성원에 24 개의 요소가 있습니다. 이 문자열 구조에 24 인 사용자 지정 길이가 있습니다.

래더 다이어그램



ST(스트럭쳐드 텍스트)

SIZE(string_a.[0].DATA[0],0,data_size_a);

추가 참조

파일/기타 명령어 페이지의 545

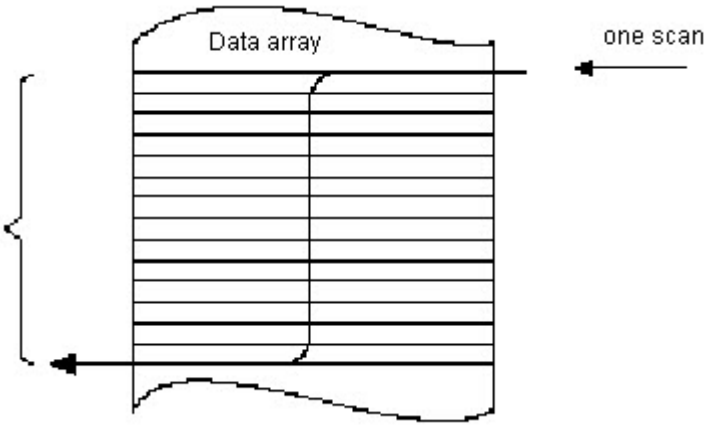
배열을 통한 인덱스 페이지의 978

데이터 변환 페이지의 967

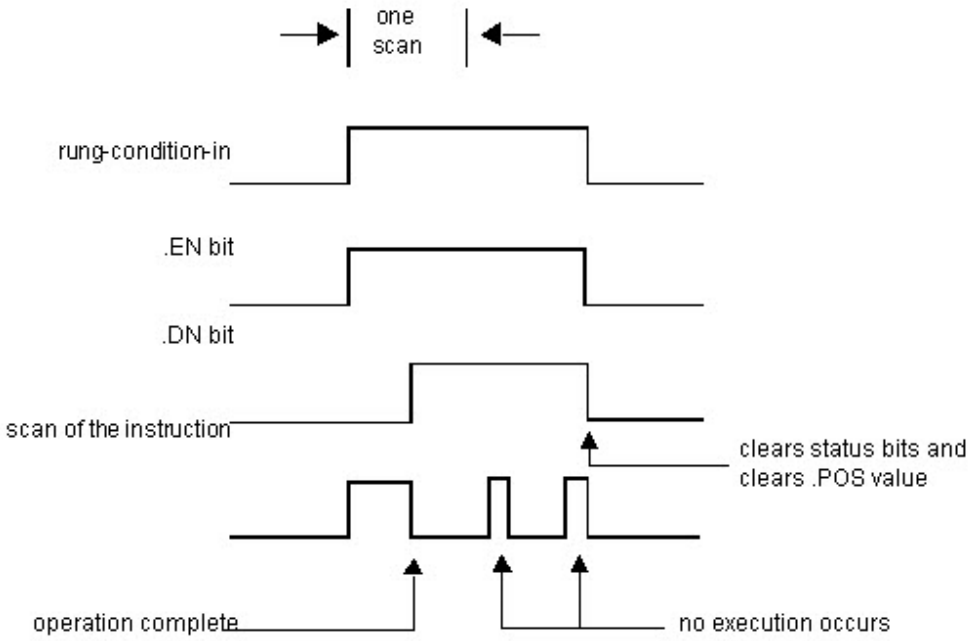
ST(스트럭쳐드 텍스트) 구문 페이지의 997

전체 모드

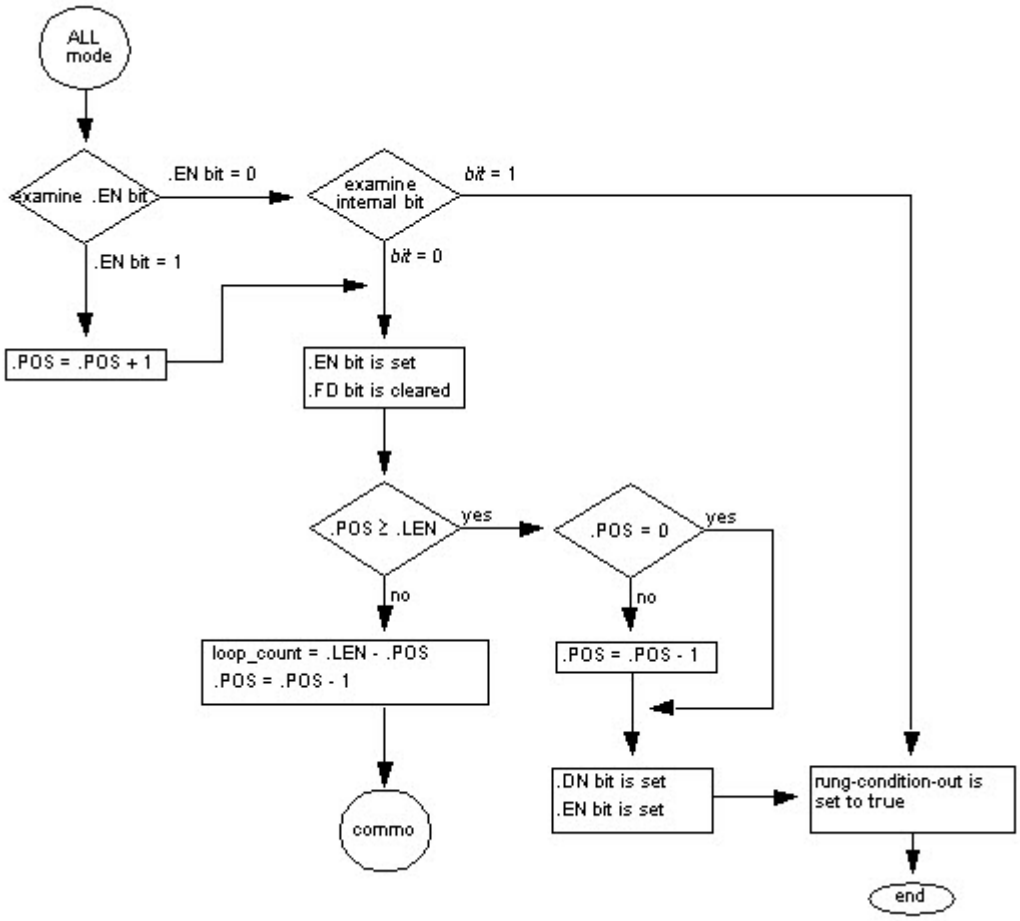
전체 모드에서는 다음 명령어로 계속 실행하기 전에 배열 내 모든 지정된 요소에서 작업합니다. 작업은 명령어의 렁-입력-조건이 거짓에서 참으로 바뀔 때 시작합니다. 제어 구조의 위치(.POS) 값은 명령어가 현재 사용하고 있는 배열 내 요소를 가리킵니다..POS 값이 .LEN 값과 같아지면 작업이 중지됩니다.



아래 타이밍 다이어그램에서는 상태 비트와 명령어 작업의 관계를 보여줍니다. 명령어 실행이 완료되면 .DN 비트가 설정됩니다. 렁-입력-조건이 거짓이면 .DN 비트, .EN 비트, .POS 값이 지워집니다. 이 경우에만 렁-입력-조건의 거짓 - 참 전환에서 명령어의 다른 실행이 트리거될 수 있습니다.



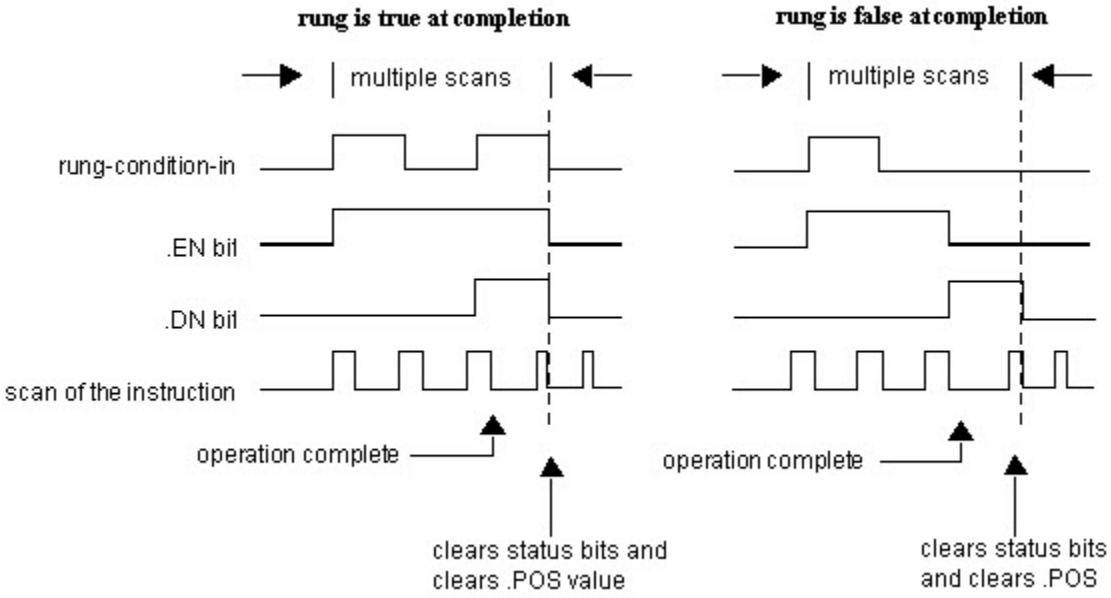
전체 모드 흐름도(FSC)



숫자 모드

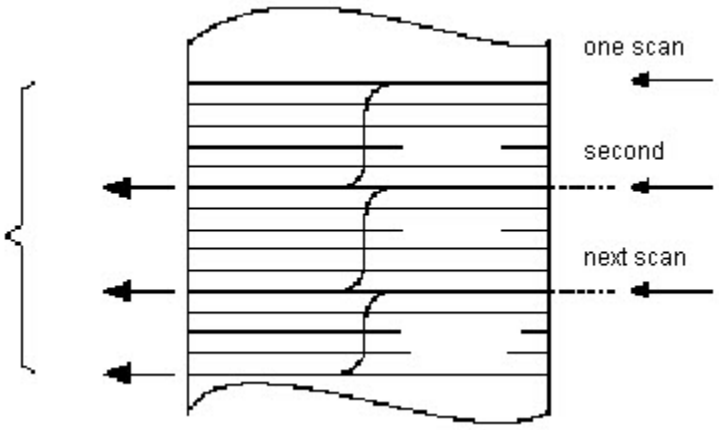
숫자 모드는 배열 작업을 여러 스캔 작업에 분배합니다. 이 모드는 비시간 임계 데이터나 대량의 데이터를 작업할 경우에 유용합니다. 스캔마다 작업할 요소의 수를 입력하여 스캔 시간을 줄일 수 있습니다.

렁-입력-조건이 거짓에서 참으로 바뀌면 실행이 트리거됩니다. 이 경우 명령어는 전체 배열의 작업을 완료하는 데 필요한 스캔 작업의 횟수 만큼 스캔될 때마다 실행됩니다. 트리거되면 렁-입력-조건은 명령어의 실행을 중단하지 않고도 반복해서 변경할 수 있습니다.



.DN 비트가 설정될 때까지 숫자 모드에서 작동되는 파일 명령어의 결과를 사용하지 마십시오.

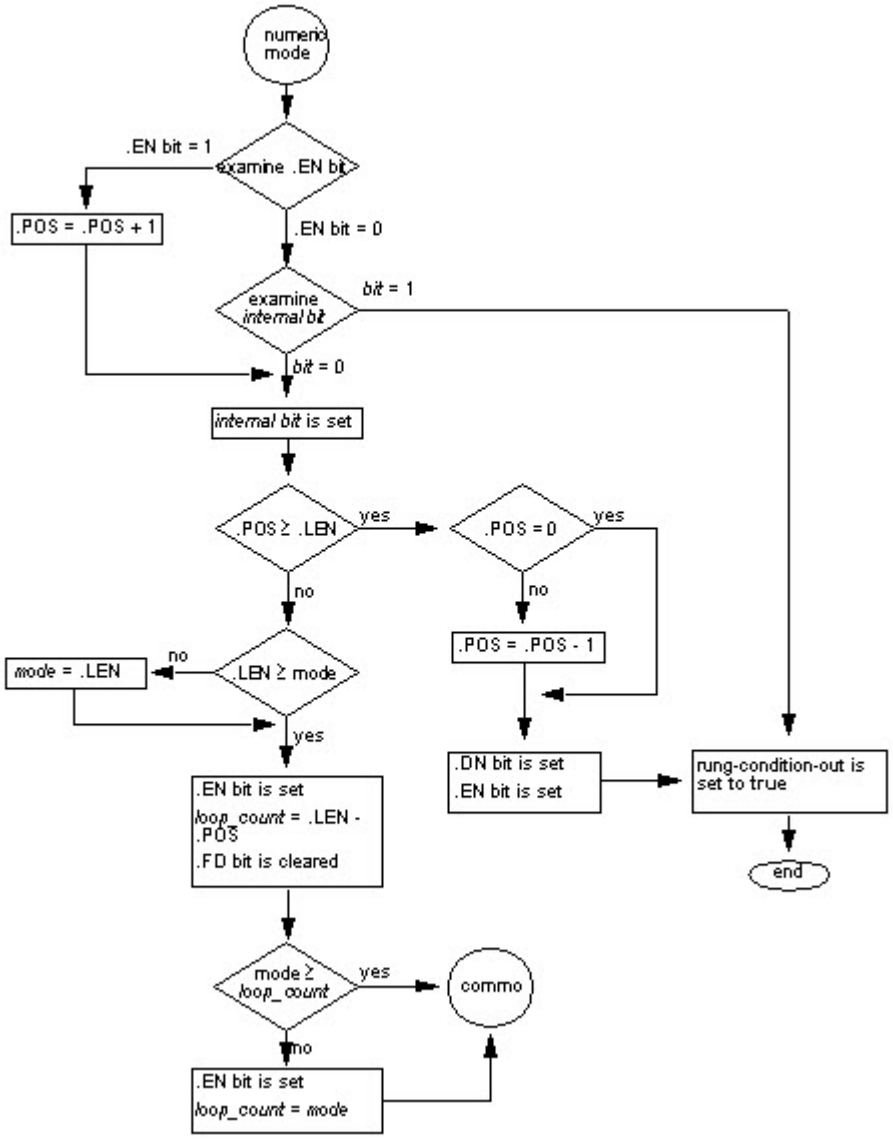
아래 타이밍 다이어그램에서는 상태 비트와 명령어 작업의 관계를 보여줍니다. 명령어 실행이 완료되면 .DN 비트가 설정됩니다.



완료 시에 렁-입력-조건이 참이면 렁-입력-조건이 거짓으로 바뀔 때까지 .EN 과 .DN 비트가 설정됩니다. 렁-입력-조건이 거짓으로 바뀌면 이러한 비트와 .POS 값은 지워집니다.

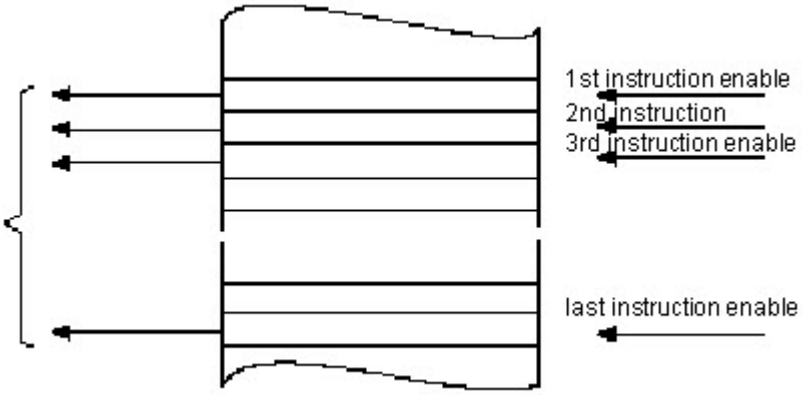
완료 시에 렁-입력-조건이 거짓이면 .EN 비트는 즉시 지워집니다. .DN 비트와 .POS 값은 .EN 비트가 지워지고 스캔이 한 번 실행된 후에 지워집니다.

숫자 모드 흐름도(FSC)

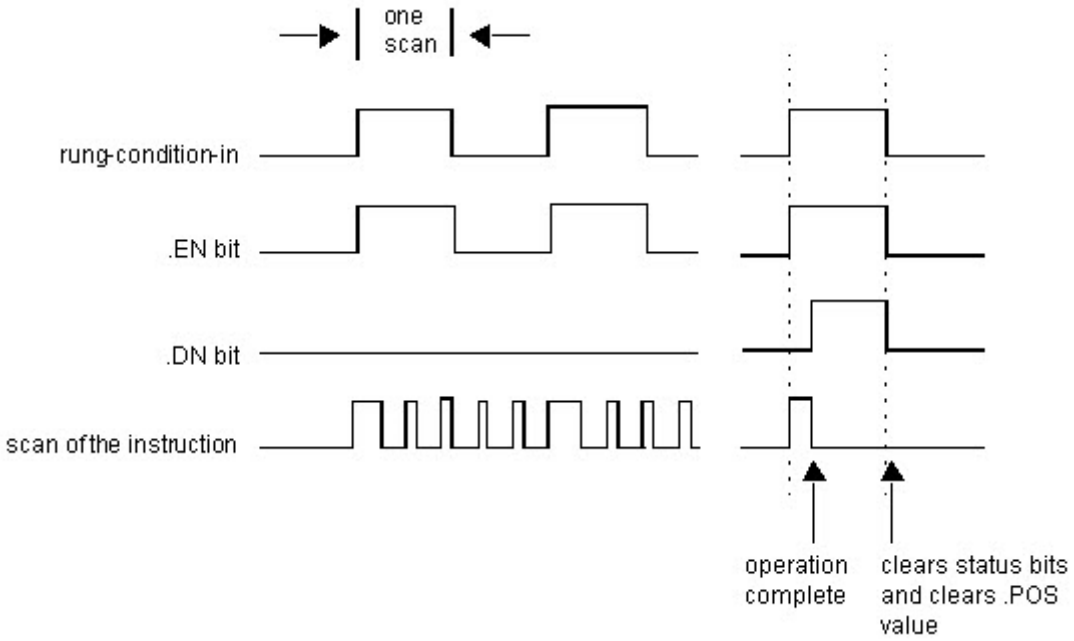


증가 모드

증가 모드는 명령어의 렁-입력-조건이 거짓에서 참으로 바뀔 때마다 배열 내 요소 하나를 조작합니다.



아래 타이밍 다이어그램에서는 상태 비트와 명령어 작업의 관계를 보여줍니다. 실행은 렁-입력-조건이 거짓에서 참으로 바뀌는 스캔에서만 진행됩니다. 이 상황이 발생할 때마다 배열 내 요소 하나만 조작됩니다. 렁-입력-조건이 둘 이상의 스캔 작업에서 계속 참인 경우, 명령어는 첫 번째 스캔 작업 중에만 실행됩니다.

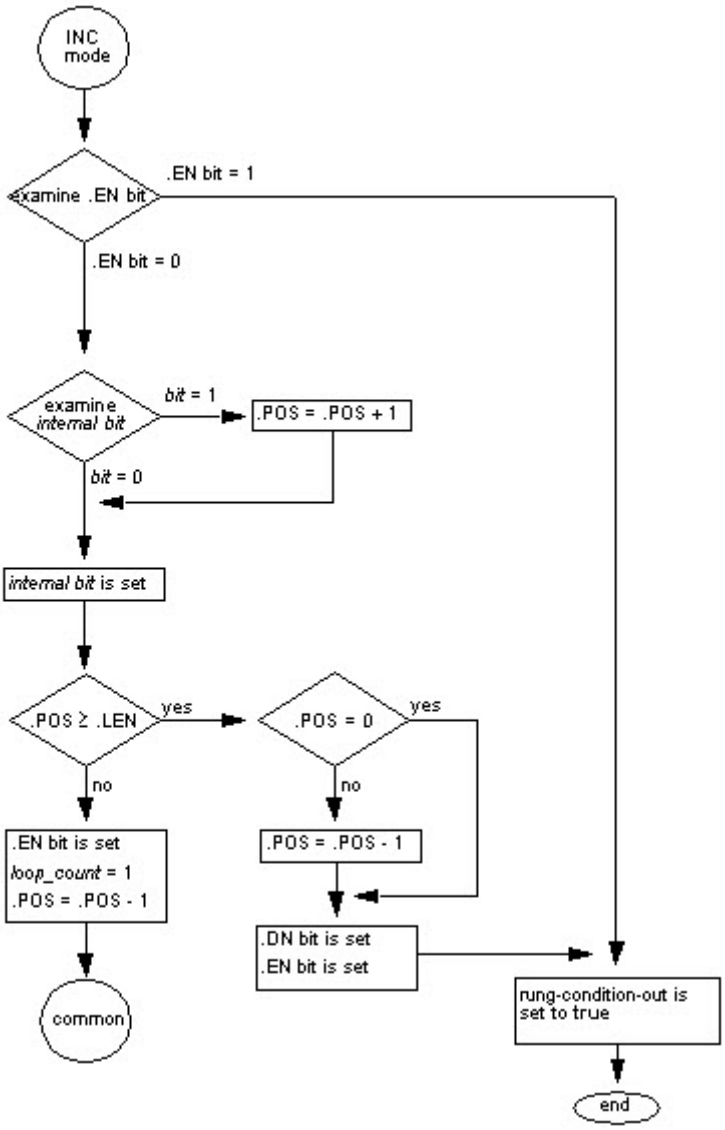


렁-입력-조건이 참인 경우, .EN 비트가 설정됩니다. 배열 내 마지막 요소가 조작되면 .DN 비트가 설정됩니다. 마지막 요소가 조작되고 렁-입력-조건이 거짓으로 전환되면 .EN 비트, .DN 비트, .POS 값이 지워집니다.

스캔당 요소 하나의 속도에 따른 증가 모드와 숫자 모드의 차이점은 다음과 같습니다.

- 스캔당 요소가 몇 개 있는 렁-입력-조건의 실행을 시작하려면 거짓 - 참 전환이 하나 있으면 됩니다. 명령어는 렁-입력-조건의 상태에 상관 없이 완료할 때까지 스캔마다 지정된 수의 요소를 계속 실행합니다.
- 증가 모드가 배열 내 요소 하나를 조작하려면 렁-입력-조건이 거짓에서 참으로 변경되어야 합니다.

증가 모드 흐름도(FSC)



배열 태그

배열 태그를 입력할 경우 배열에서 조작할 첫 번째 요소를 지정해야 합니다. 이 명령어가 작동할 때 .POS 값을 변경해 결과가 손상될 수 있으므로 CONTROL.POS 를 사용하여 시작 요소를 식별하지 마십시오.

표준 편차

표준 편차는 다음 공식에 따라 계산됩니다.

$$\text{Standard Deviation} = \sqrt{\frac{\left(\sum_{i=1}^{N}\left[\langle X_{(start+i)} - AVE\rangle^2\right]\right)}{(N-1)}}$$

여기에서:

- start = 배열 피연산자의 dimension-to-vary 첨자
- xi = 배열의 변수 요소
- N = 배열의 지정된 요소 수
- AVE = $\frac{\left(\sum_{i=1}^{N} x_{(start+i)}\right)}{N}$

배열(파일)/이동 명령어

배열(파일)/이동 명령어

배열(파일)/이동 명령어를 사용하여 배열 내의 데이터 위치를 수정할 수 있습니다.

사용 가능한 명령어

래더 다이어그램

BSL	BSR	FFL	FFU	LFL	LFU

평션 블록

사용할 수 없음

ST(스트럭쳐드 텍스트)

사용할 수 없음

실행할 작업:	사용할 명령어:
비트 배열에서 한 번에 한 개의 비트를 로드하고, 이동하고, 언로드합니다.	BSL BSR
값을 동일한 순서로 로드하고 언로드합니다.	FFL FFU
값을 역순으로 로드하고 언로드합니다.	LFL LFU

데이터 유형을 혼합 사용할 수 있지만 정확도가 떨어지고 반올림 에러가 발생할 수 있습니다.

볼드체 데이터 유형은 최적의 데이터 유형을 나타냅니다. 명령어의 모든 피연산자가 동일한 최적의 데이터 유형(일반적으로 DINT 또는 REAL)을 사용하는 경우 명령어는 가장 적은 메모리를 사용하면서 보다 빠르게 실행됩니다.

추가 참조

ASCII 변환 명령어 페이지의 927

ASCII 시리얼 포트 명령어 페이지의 907

ASCII 문자열 명령어 페이지의 907

비트 왼쪽 자리 이동(BSL)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

BSL 명령어가 배열 내에서 지정된 비트를 왼쪽으로 한 위치 옮깁니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

래더 다이어그램

피연산자	유형	형식	설명
배열(Array)	DINT ARRAY	태그	수정할 배열 이동을 시작할 첫 번째 요소를 지정합니다.
제어(Control)	CONTROL	태그	연산의 제어 구조
소스 비트(Source Bit)	BOOL	태그	비어 있는 위치로 이동할 비트
길이(Length)	DINT	즉시	배열에서 이동할 비트의 수

CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 BSL 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 비트가 왼쪽으로 한 위치 이동했음을 나타내도록 설정됩니다.
.UL	BOOL	언로드 비트는 명령어의 출력입니다. .UL 비트가 비트 범위 밖으로 옮겨진 비트의 상태를 저장합니다.
.ER	BOOL	에러 비트는 .LEN < 0 인 경우에 설정됩니다.
.LEN	DINT	길이는 이동할 배열 비트의 수를 지정합니다.

설명

활성화된 경우 이 명령어는 지정된 비트의 맨 위에 있는 비트를 .UL 비트에 언로드하고, 나머지 비트를 왼쪽으로 한 위치 옮긴 다음 비트 주소를 Array 의 비트 0 에 로드합니다.

중요: 명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

BSL 명령어는 연속 데이터 메모리에서 작동됩니다. BSL 명령어는 연속 데이터 메모리에서 작동됩니다. CompactLogix 5370 및 ControlLogix 5570 컨트롤러의 경우에만 명령어의 범위가 기본 태그로 제한됩니다. BSL 명령어는 기본 태그의 외부에 데이터를

쓰지 않지만 구성원 경계를 넘을 수 있습니다. 구조의 구성원인 배열을 지정하고 길이가 해당 배열의 크기를 초과할 경우 BSL 명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에서는 데이터가 지정된 구성원으로 제한됩니다.

이 전환 명령어에서 명령어가 실행되려면 릴레이 래더가 렁-입력-조건을 거짓에서 참으로 토글 전환합니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트가 발생하는 조건	폴트 유형	폴트 코드
LEN 이 배열의 크기를 초과합니다	4	20

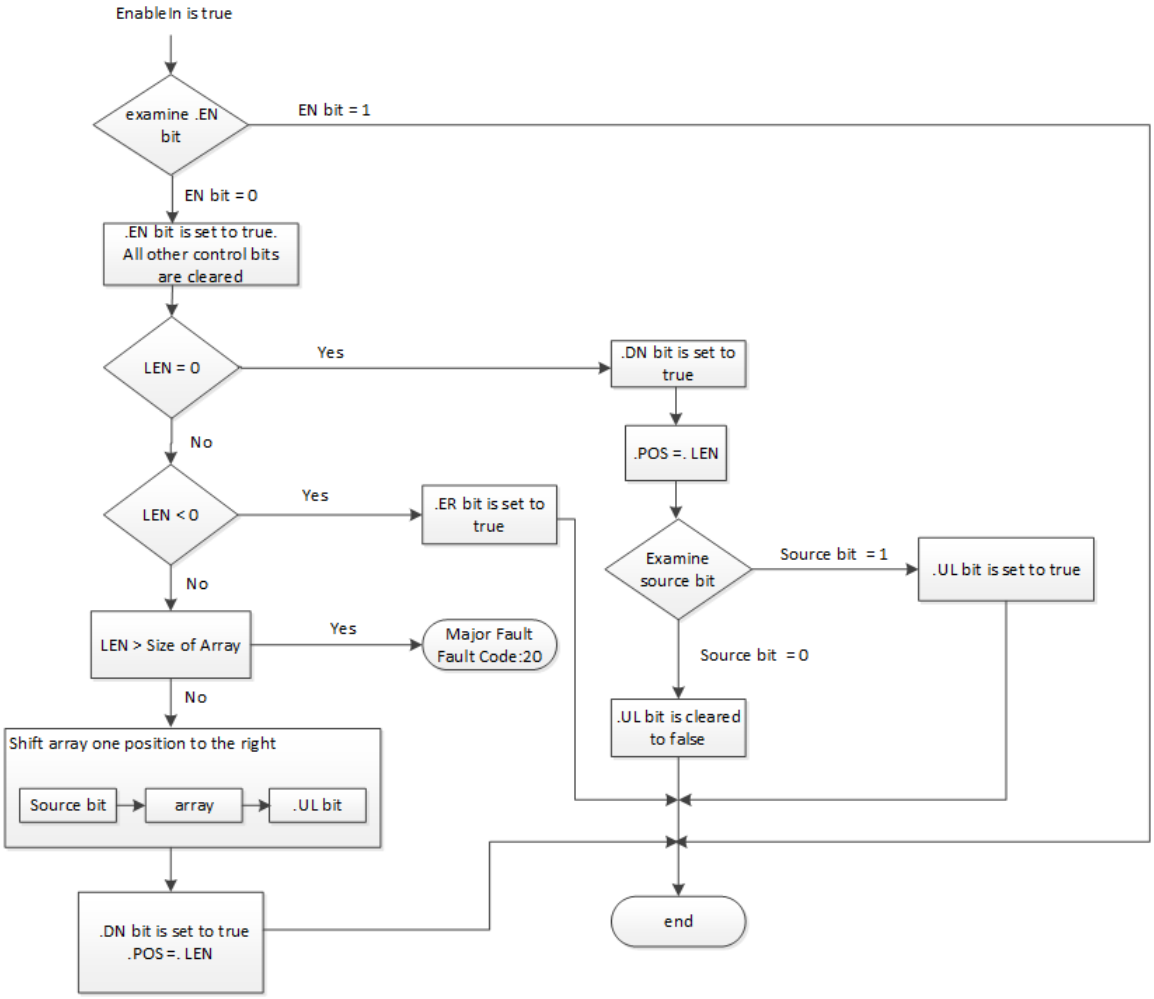
피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	.EN 비트가 거짓으로 해제됩니다. .DN 비트가 거짓으로 해제됩니다. .ER 비트가 거짓으로 해제됩니다. .POS 값이 해제됩니다.
렁-입력-조건이 거짓임	.EN 비트가 거짓으로 해제됩니다. .DN 비트가 거짓으로 해제됩니다. .ER 비트가 거짓으로 해제됩니다. .POS 값이 해제됩니다.
렁-입력-조건이 참임	BSL 흐름도(참)를 참조하십시오.
사후 스캔	N/A

BSL 흐름도(참)

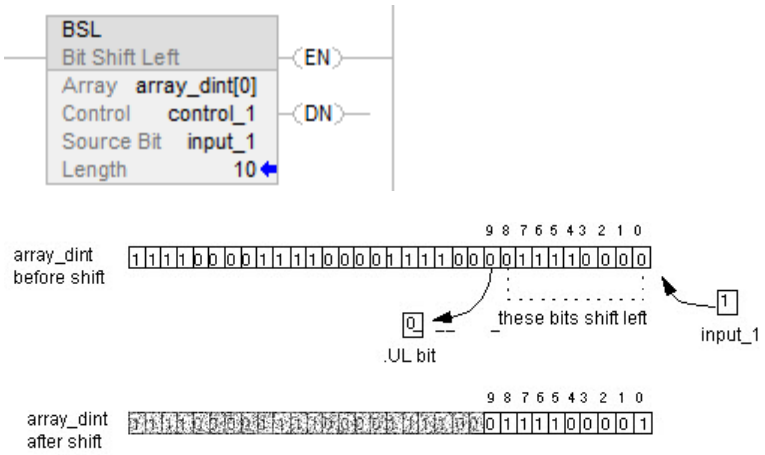


예제

예 1

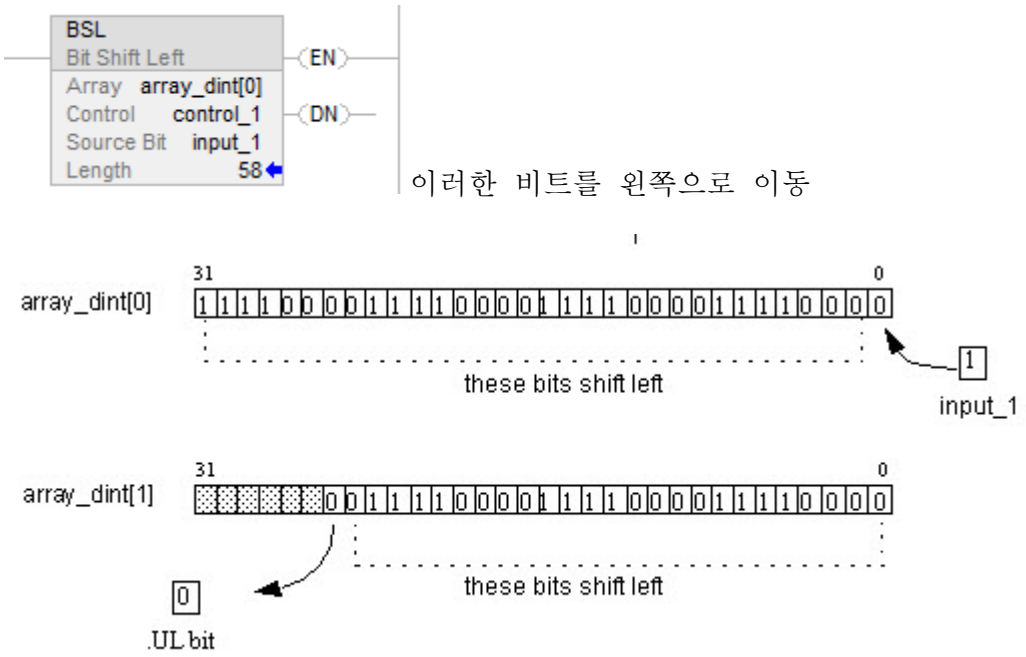
활성화된 경우 BSL 명령어는 array_dint[0]의 비트 0 에서 시작합니다. 이 명령어는 array_dint[0].9 를 .UL 비트에 언로드하고, 나머지 비트를 이동한 다음 input_1 을 array_dint[0].0 에 로드합니다. 나머지 비트(10 ~ 31)가 유효하지 않습니다.

래더 다이어그램



예 2:

활성화된 경우 BSL 명령어는 array_dint[0]의 비트 0 에서 시작합니다. 이 명령어는 array_dint[1].25 를 .UL 비트에 언로드하고, 나머지 비트를 이동한 다음 input_1 을 array_dint[0].0 에 로드합니다. 나머지 비트(array_dint[1]의 31 ~ 26)가 유효하지 않습니다.



추가 참조

공통 속성 페이지의 963

데이터 변환 페이지의 967

비트 오른쪽 자리 이동(BSR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

BSR 명령어가 배열 내에서 지정된 비트를 오른쪽으로 한 위치 옮깁니다. 활성화된 경우 이 명령어는 배열의 비트 0 의 값을 .UL 비트에 언로드하고, 나머지 비트를 오른쪽으로 한 위치 옮긴 다음 비트 주소에서 비트를 로드합니다.

중요:	명령어가 올바른 데이터를 변경했는지 테스트하고 확인하십시오. BSR 명령어는 연속 메모리에서 작동됩니다. Array 가 구성원 배열인 경우 이 명령어가 배열의 경계를 넘어서 구성원의 뒤에 오는 다른 구성원으로 이동할 수 있습니다. 이러한 상황이 발생하지 않도록 주의해서 길이를 선택하십시오.

BSR 명령어는 연속 데이터 메모리에서 작동됩니다. CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러의 경우에만 명령어의 범위가 기본 태그로 제한됩니다. BSL 명령어는 기본 태그의 외부에 데이터를 쓰지 않지만 구성원 경계를 넘을 수 있습니다. 구조의 구성원인 배열을 지정하고 길이가 해당 배열의 크기를 초과할 경우 BSL 명령어가 올바른 데이터를 변경했는지 테스트하고 확인해야 합니다.

CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러에서는 데이터가 지정된 구성원으로 제한됩니다.

이 명령어가 배열의 끝을 지나서 읽기 작업을 수행하려고 할 경우(LEN 이 너무 큼) .ER 비트를 설정하여 메이저 폴트가 발생합니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	데이터 유형	형식	설명
Array	DINT ARRAY	태그	수정할 배열 이동할 첫 번째 요소를 지정합니다.
Control	CONTROL	태그	연산의 제어 구조
Source Bit	BOOL	태그	비어 있는 위치에 로드할 비트.
Length	DINT	즉시	배열에서 이동할 비트의 수

CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 BSR 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 비트가 오른쪽으로 한 위치 이동했음을 나타내도록 설정됩니다.
.UL	BOOL	언로드 비트는 명령어의 출력입니다. .UL 비트가 비트 범위 밖으로 옮겨진 비트의 상태를 저장합니다.
.ER	BOOL	에러 비트는 .LEN < 0 인 경우에 설정됩니다.
.LEN	DINT	길이는 이동할 배열 비트의 수를 지정합니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

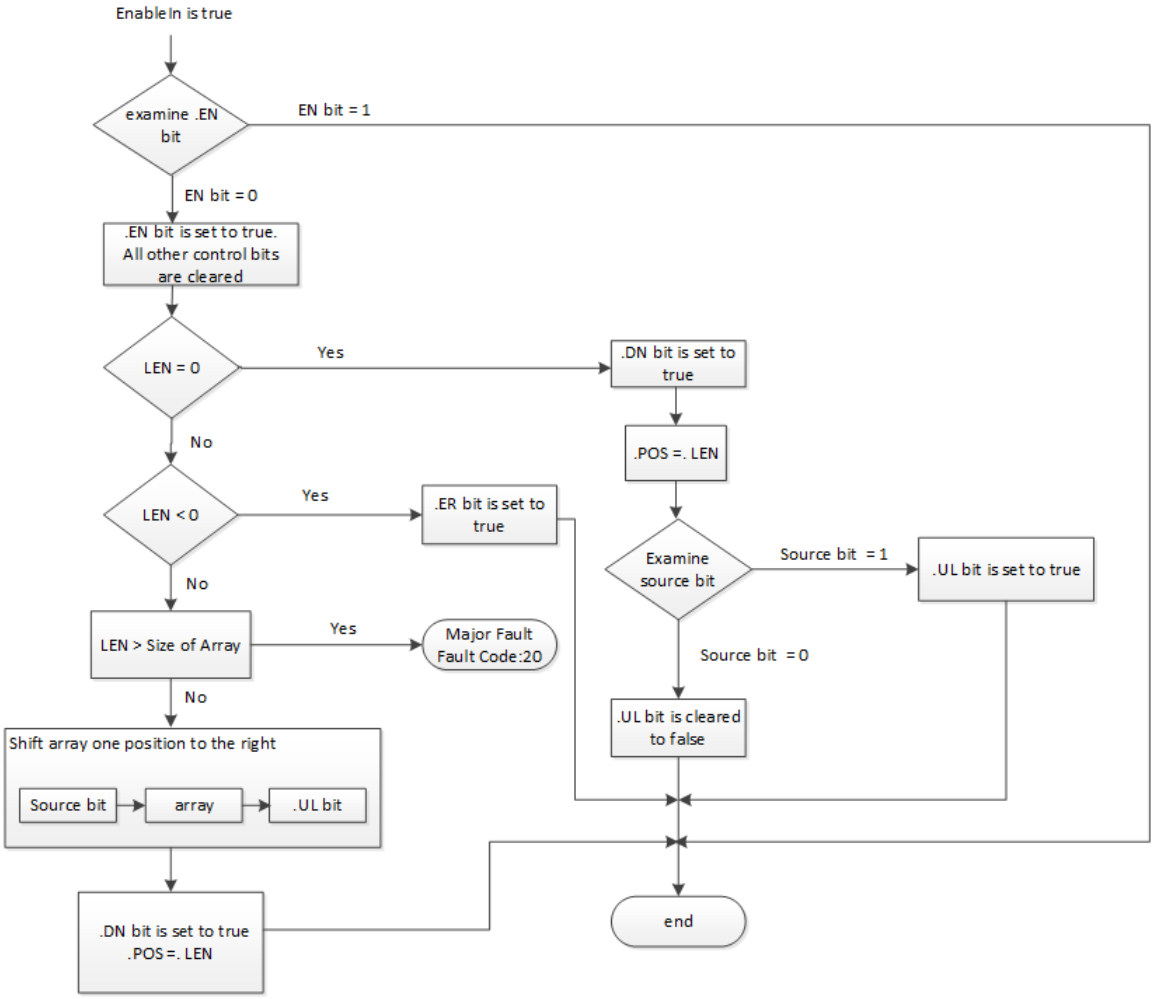
이 명령어에만 해당되는 것은 없습니다. 배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	.EN 비트가 거짓으로 해제됩니다. .DN 비트가 거짓으로 해제됩니다. .ER 비트가 거짓으로 해제됩니다. .POS 값이 해제됩니다.
렁-입력-조건이 거짓임	.EN 비트가 거짓으로 해제됩니다. .DN 비트가 거짓으로 해제됩니다. .ER 비트가 거짓으로 해제됩니다. .POS 값이 해제됩니다.
렁-입력-조건이 참임	다음 BSR 흐름도(참)를 참조하십시오.
사후 스캔	N/A

BSR 흐름도(참)

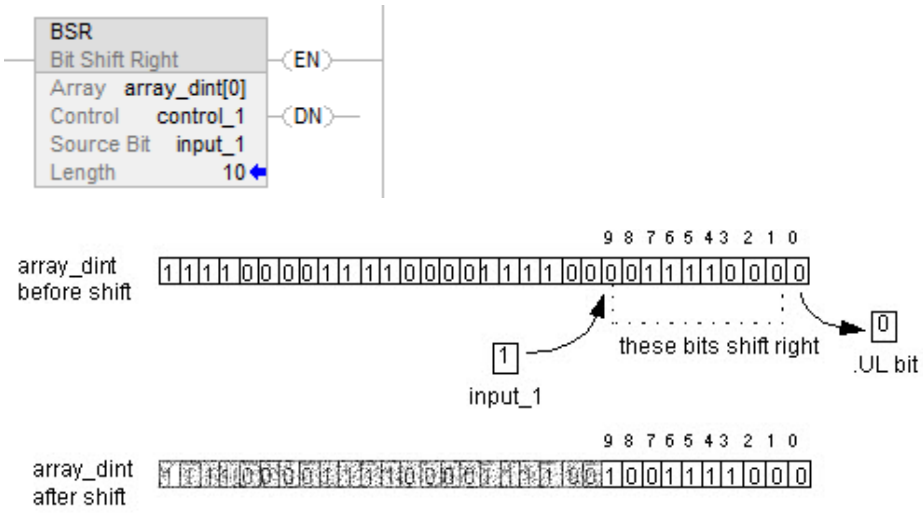


예

예 1

활성화된 경우 BSR 명령어는 array_dint[0].0 을 .UL 비트에 복사하고, 0 ~ 9 를 오른쪽으로 이동한 다음 input_1 을 array_dint[0].9 에 로드합니다. 나머지 비트(10 ~ 31)가 유효하지 않습니다. 비트가 수정되지 않았을 수 있습니다.

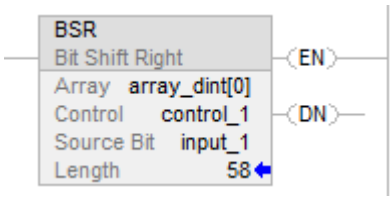
래더 다이어그램



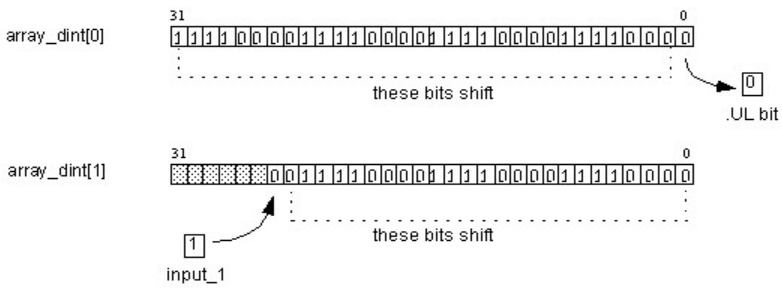
예 2

활성화된 경우 BSR 명령어는 array_dint[0].0 을 .UL 비트에 복사하고, 0 ~ 9 를 오른쪽으로 이동한 다음 input_1 을 array_dint[1].25 에 로드합니다. 나머지 비트(dint_array[1]의 31 ~ 26)가 유효하지 않습니다. 비트가 수정되지 않았을 수 있습니다. array_dint[1].0 이 워드를 넘어서 array_dint[0].31 로 이동하는 방식을 주의 깊게 살펴보십시오.

래더 다이어그램



이러한 비트를 오른쪽으로 이동



추가 참조

배열을 통한 인덱스 페이지의 978

데이터 변환 페이지의 967

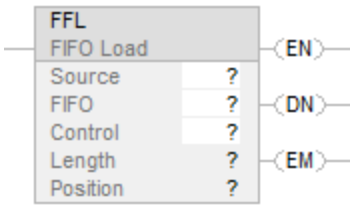
FIFO 로드(FFL)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

FFL 명령어는 Source 값을 FIFO 에 복사하는 데 사용됩니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

소스 피연산자의 유형이 FIFO 의 유형과 일치하지 않는 경우에만 변환이 발생합니다.

래더 다이어그램

피연산자	유형	형식	설명
소스(Source)	SINT INT DINT REAL 문자열 유형 구조	즉시 태그	FIFO 에 저장할 데이터.
FIFO	SINT INT DINT REAL 문자열 유형 구조	배열 태그	수정할 FIFO. FIFO 의 첫 번째 요소를 지정합니다.
제어(Control)	CONTROL	태그	연산의 제어 구조 일반적으로 연결된 FFU 와 동일한 CONTROL 을 사용합니다.
길이(Length)	DINT	즉시	FIFO 가 한 번에 수용할 수 있는 최대 요소 수
위치(Position)	DINT	즉시	명령어가 데이터를 로드할 FIFO 의 다음 위치 초기값은 일반적으로 0 입니다.

CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 FFL 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 FIFO 가 가득 찼음을 나타내도록 설정됩니다. .DN 비트가 .POS < .LEN 이 될 때까지 FIFO 를 로드하지 못하게 합니다.
.EM	BOOL	빈 비트는 FIFO 가 비어 있음을 나타냅니다. .LEN < 0, .LEN = 0 또는 .POS < 0 이면 .EM 비트와 .DN 비트가 설정됩니다.
.LEN	DINT	길이 워드는 FIFO 의 최대 요소 수를 지정합니다.
.POS	DINT	위치 워드는 FIFO 에서 명령어가 다음 값을 로드할 위치를 식별합니다.

설명

FFL 명령어와 FFU 명령어를 함께 사용하여 선입선출의 순서로 데이터를 저장하고 검색할 수 있습니다. FFL 명령어와 FFU 명령어를 한 쌍으로 사용할 경우 비동기식 이동 레지스터가 구성됩니다.

일반적으로 Source 와 FIFO 의 데이터 유형이 동일합니다.

활성화된 경우 FFL 명령어는 FIFO 에서 .POS 값으로 식별한 위치에 Source 값을 로드합니다. 이 명령어는 FIFO 가 가득 찰 때까지 활성화될 때마다 값을 한 개씩 로드합니다.

중요:	명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

FFL 명령어는 연속 메모리에서 작동됩니다. BSL 명령어는 연속 데이터 메모리에서 작동됩니다. CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러의 경우에만 명령어의 범위가 기본 태그로 제한됩니다. BSL 명령어는 기본 태그의 외부에 데이터를 쓰지 않지만 구성원 경계를 넘을 수 있습니다. 구조의 구성원인 배열을 지정하고 길이가 해당 배열의 크기를 초과할 경우 BSL 명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에서는 데이터가 지정된 구성원으로 제한됩니다.

이 명령어가 배열의 끝을 지나서 읽기 작업을 수행하려고 할 경우 메이저 폴트가 발생합니다.

일반적으로 Source 와 FIFO 의 데이터 유형이 동일합니다. Source 와 FIFO 의 데이터 유형이 일치하지 않을 경우 이 명령어가 Source 값을 FIFO 태그의 데이터 유형으로 변환합니다.

작은 정수는 부호 확장을 통해 큰 정수로 변환됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
(시작 요소 + .POS)가 FIFO 배열의 끝을 지난 경우	4	20

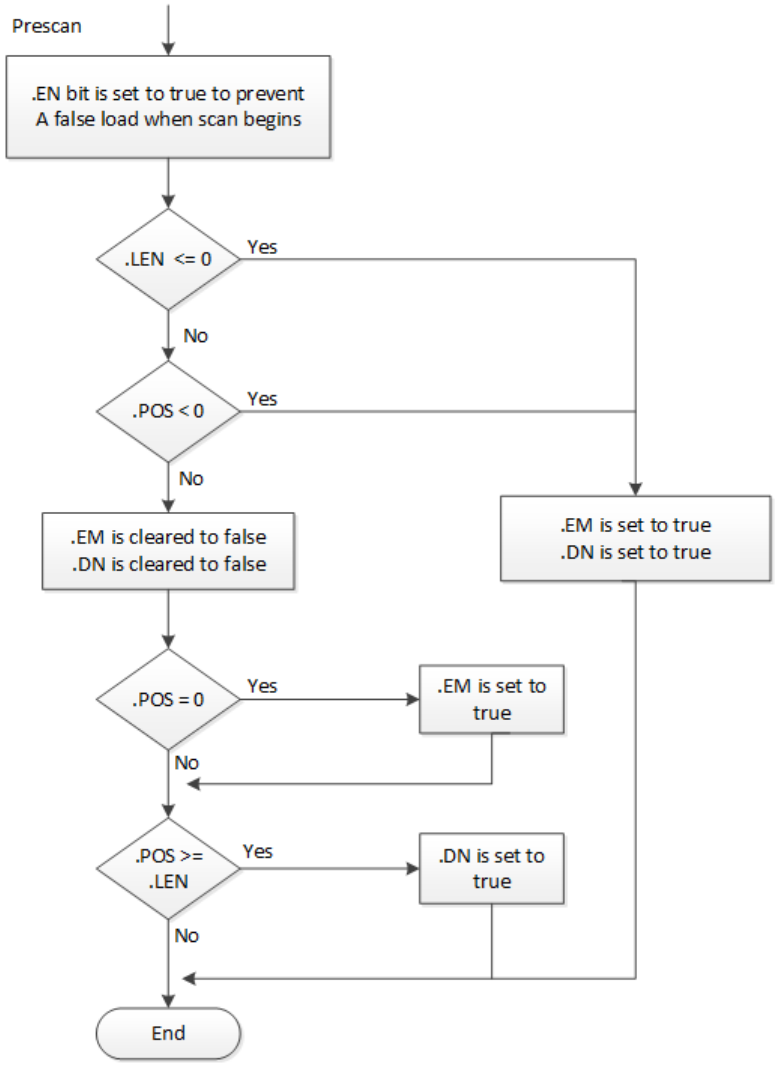
피연산자 관련 폴트에 대해서는 공통 특성을 참조하십시오.

실행

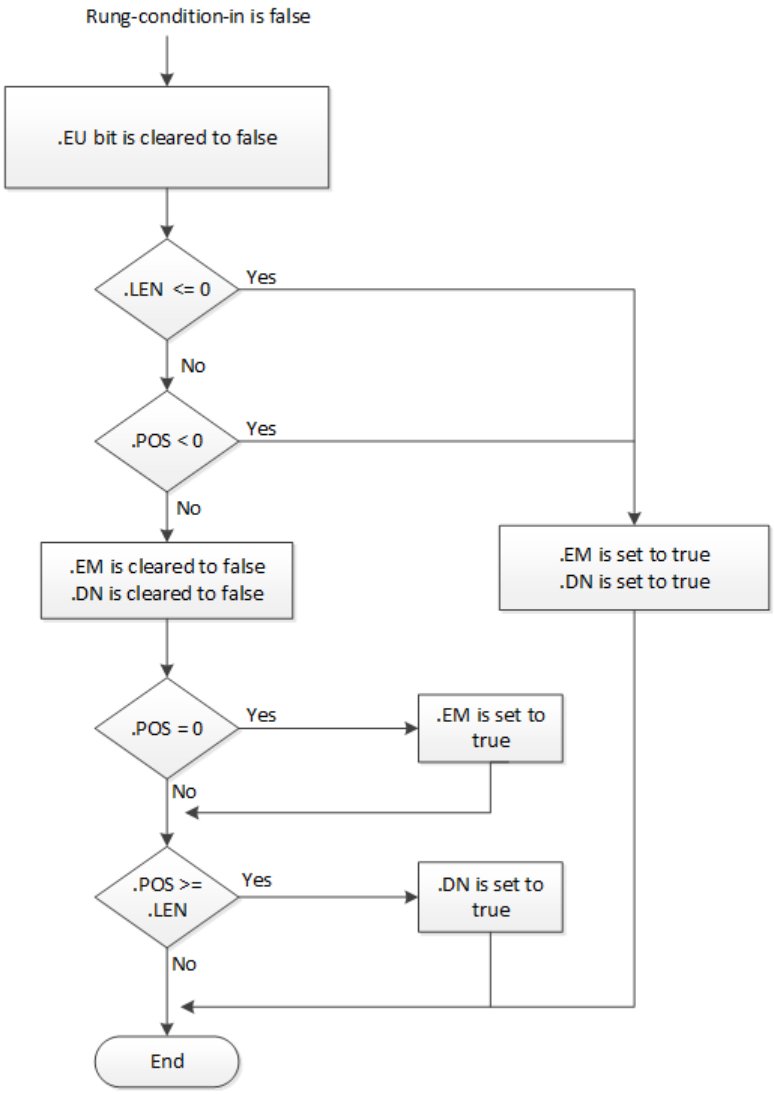
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	FFL 흐름도(사전 스캔)를 참조하십시오.
렁-입력-조건이 거짓임	FFL 흐름도(거짓)를 참조하십시오.
렁-입력-조건이 참임	FFL 흐름도(참)를 참조하십시오.
사후 스캔	N/A

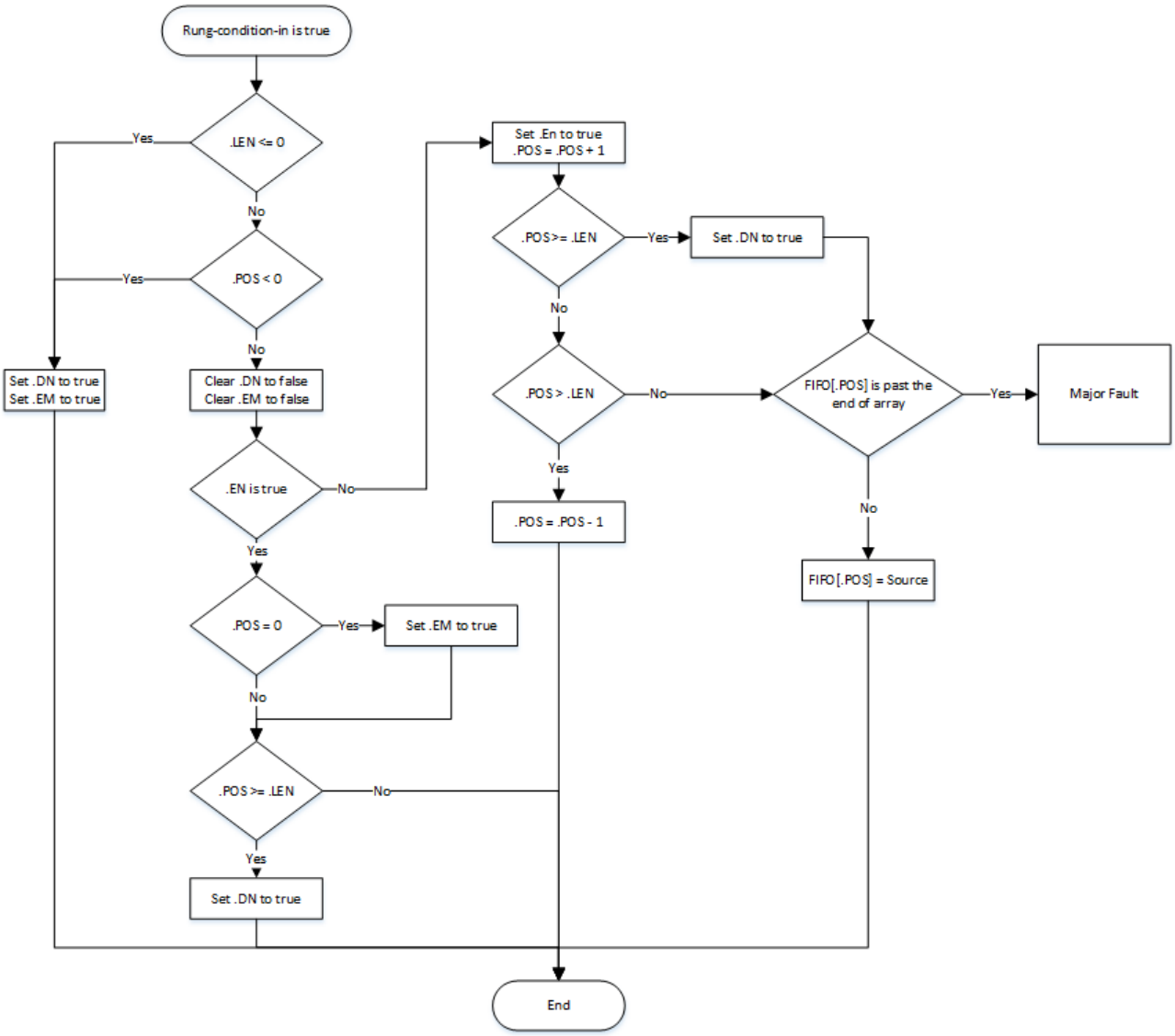
FFL 흐름도(사전 스캔)


Prescan
.EN bit is set to true to prevent
A false load when scan begins
.LEN <= 0
Yes
No
.POS < 0
Yes
No
.EM is cleared to false
.DN is cleared to false
.EM is set to true
.DN is set to true
.POS = 0
Yes
.EM is set to true
No
.POS >= .LEN
Yes
.DN is set to true
No
End

FFL 흐름도(거짓)


Rung-condition-in is false
.EU bit is cleared to false
.LEN <= 0
Yes
No
.POS < 0
Yes
No
.EM is cleared to false
.DN is cleared to false
.EM is set to true
.DN is set to true
.POS = 0
Yes
.EM is set to true
No
.POS >= .LEN
Yes
.DN is set to true
No
End

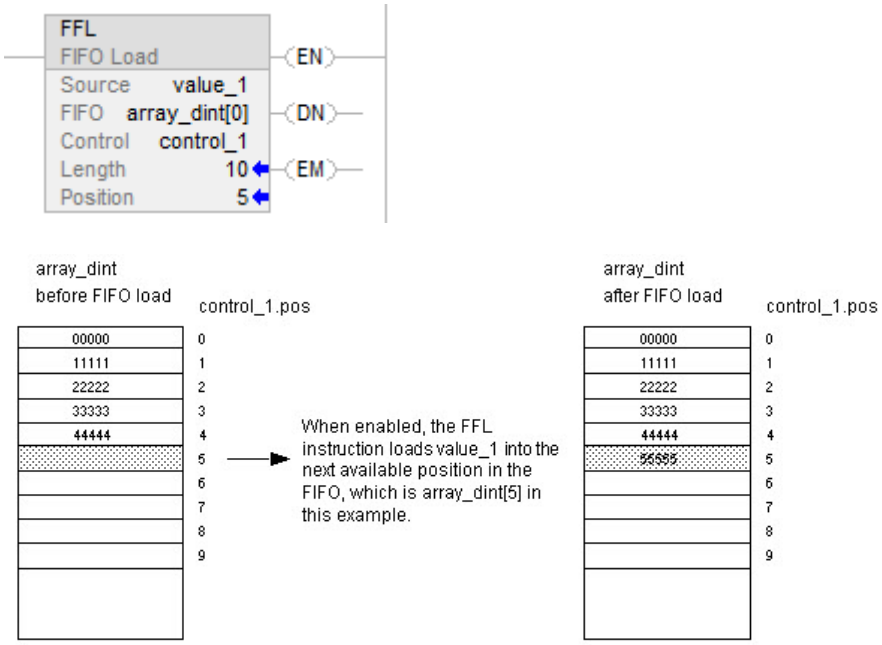
FFL 흐름도(참)


Rung-condition-in is true
.LEN <= 0
Yes
No
.POS < 0
Yes
No
Set .DN to true
Set .EM to true
Clear .DN to false
Clear .EM to false
.EN is true
No
Yes
.POS = 0
Yes
Set .EM to true
No
.POS >= .LEN
No
Yes
Set .DN to true
Set .En to true
.POS = .POS + 1
.POS >= .LEN
Yes
Set .DN to true
No
.POS > .LEN
No
FIFO[.POS] is past the end of array
Yes
Major Fault
Yes
.POS = .POS - 1
No
FIFO[.POS] = Source
End

예제

예 1

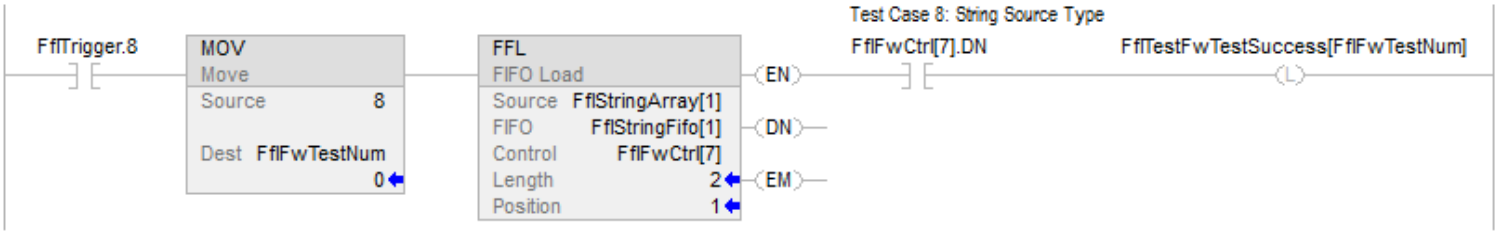
래더 다이어그램



예 2

Source 배열이 STRING 배열이거나 구조 배열입니다.

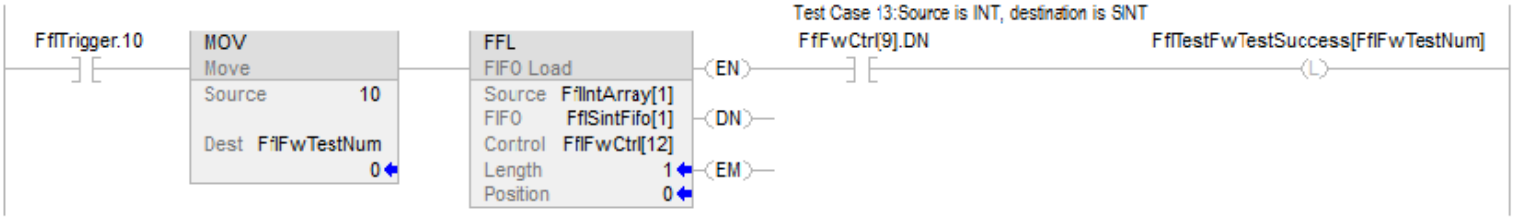
래더 다이어그램



예 3

소스의 데이터 유형이 FIFO 배열의 데이터 유형과 일치하지 않습니다.

래더 다이어그램



추가 참조

배열(파일)/이동 명령어 페이지의 619

FIFO 언로드(FFU) 페이지의 638

LIFO 로드(LFL) 페이지의 646

공통 속성 페이지의 963

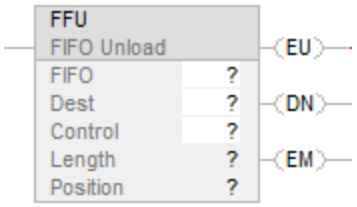
FIFO 언로드(FFU)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

FFU 명령어는 FIFO 의 위치 0(첫 번째 위치)에서 값을 언로드하고 이 값을 Destination 에 저장합니다. FIFO 의 나머지 비트는 한 위치 아래로 이동합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다.

래더 다이어그램

피연산자	유형	형식	설명
FIFO	SINT INT DINT REAL 문자열 유형 구조	배열 태그	수정할 FIFO. FIFO 의 첫 번째 요소를 지정합니다. 첨자에 CONTROL.POS 를 사용하지 마십시오.
Destination	SINT INT DINT REAL 문자열 유형 구조	태그	FIFO 에서 언로드된 값.
Control	CONTROL	태그	연산의 제어 구조 일반적으로 연결된 FFL 과 동일한 CONTROL 을 사용합니다.
Length	DINT	즉시	FIFO 가 한 번에 수용할 수 있는 최대 요소 수
Position	DINT	즉시	명령어가 데이터를 로드할 FIFO 의 다음 위치 초기값은 일반적으로 0 입니다.

CONTROL 구조

니모닉	데이터 유형	설명
.EU	BOOL	언로드 비트는 FFU 명령어가 활성화되었음을 나타냅니다. .EU 비트는 사전 스캔이 시작될 경우 거짓 언로드를 방지하도록 설정됩니다.
.DN	BOOL	완료 비트는 FIFO 가 가득 찼음을(.POS = .LEN) 나타내도록 설정됩니다.
.EM	BOOL	빈 비트는 FIFO 가 비어 있음을 나타냅니다. .LEN < 0, .LEN = 0 또는 .POS < 0 이면 .EM 비트와 .DN 비트가 설정됩니다.
.LEN	DINT	길이는 FIFO 의 최대 요소 수를 지정합니다.
.POS	DINT	위치는 FIFO 에 로드된 데이터의 끝을 식별합니다.

설명

FFU 명령어와 FFL 명령어를 함께 사용하여 선입선출의 순서로 데이터를 저장하고 검색할 수 있습니다.

활성화된 경우 FFU 명령어는 FIFO 의 첫 번째 요소에서 데이터를 언로드하고 해당 값을 Destination 에 저장합니다. 이 명령어는 FIFO 가 빌 때까지 활성화될 때마다 값을 한 개씩 언로드합니다. FIFO 가 비면 FFU 가 Destination 에 0 을 반환합니다.

일반적으로 대상과 FIFO 의 데이터 유형이 동일합니다. 데이터 유형이 다를 경우 이 명령어가 언로드된 값을 대상 태그의 유형으로 변환합니다.

작은 정수는 부호 확장을 통해 큰 정수로 변환됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
지정된 Length 가 FIFO 배열의 끝을 지남	4	20

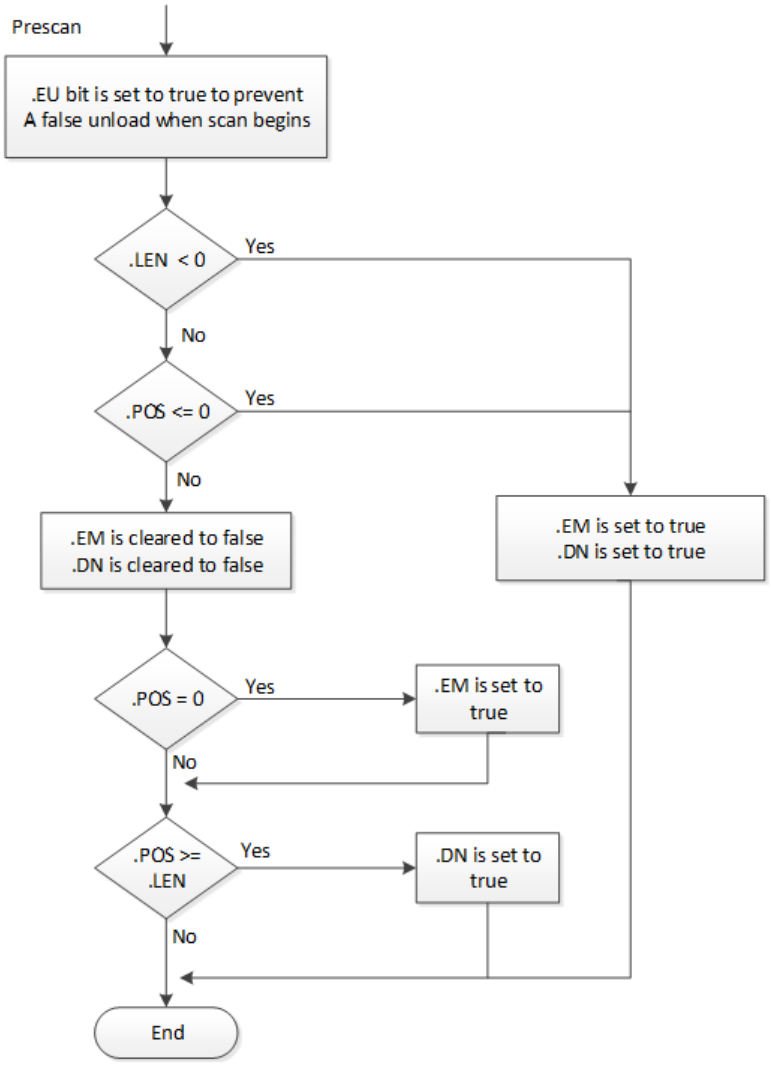
피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

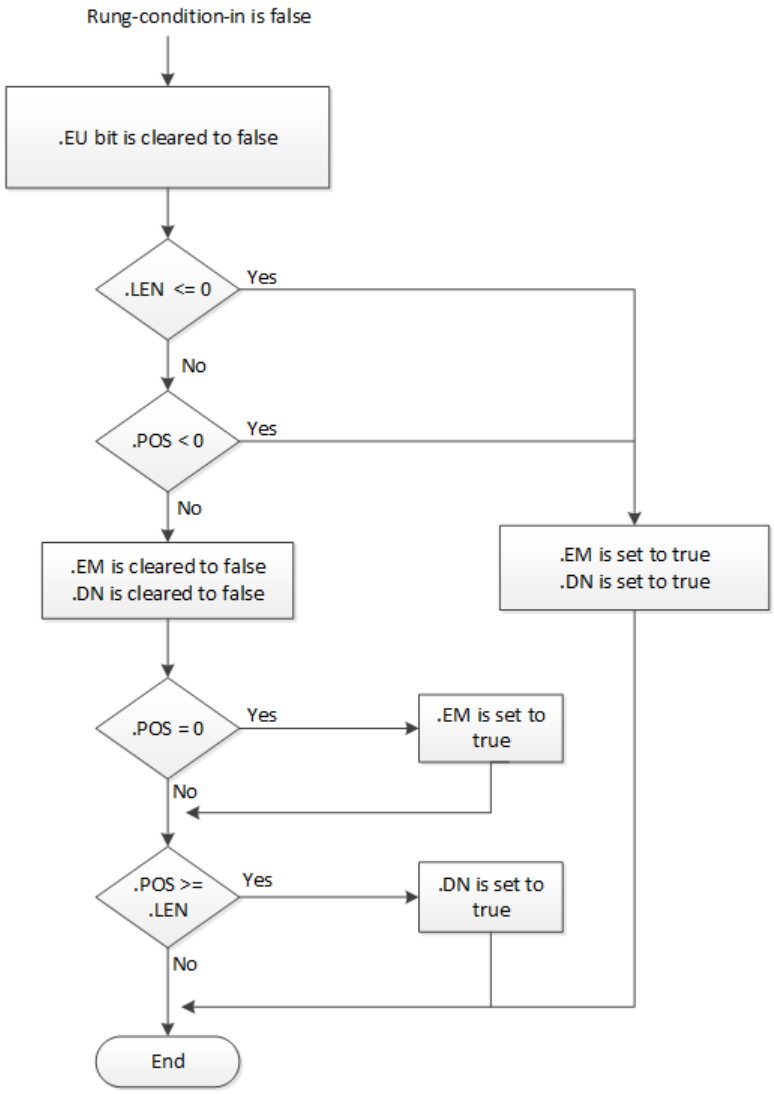
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	FFU 흐름도(사전 스캔)를 참조하십시오.
렁-입력-조건이 거짓임	FFL 흐름도(거짓)를 참조하십시오.
렁-입력-조건이 참임	FFU 흐름도(참)를 참조하십시오.
사후 스캔	N/A

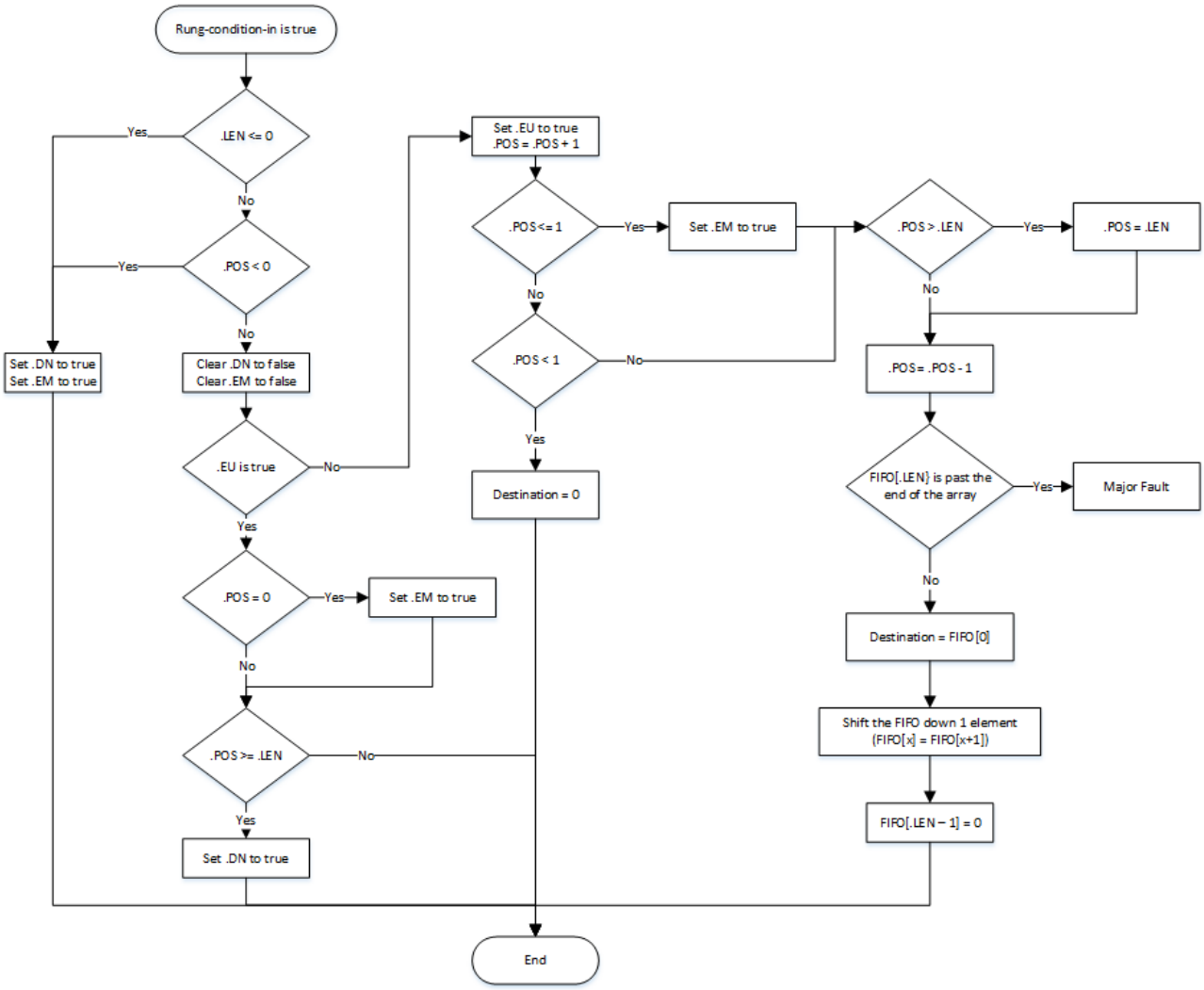
FFU 흐름도(사전 스캔)


Prescan
.EU bit is set to true to prevent
A false unload when scan begins
.LEN < 0
Yes
No
.POS <= 0
Yes
No
.EM is cleared to false
.DN is cleared to false
.EM is set to true
.DN is set to true
.POS = 0
Yes
.EM is set to true
No
.POS >= .LEN
Yes
.DN is set to true
No
End

FFL 흐름도(거짓)


Rung-condition-in is false
.EU bit is cleared to false
.LEN <= 0
Yes
No
.POS < 0
Yes
No
.EM is cleared to false
.DN is cleared to false
.EM is set to true
.DN is set to true
.POS = 0
Yes
.EM is set to true
No
.POS >= .LEN
Yes
.DN is set to true
No
End

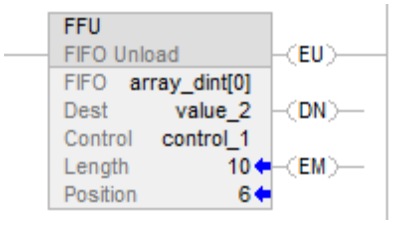
FFU 흐름도(참)

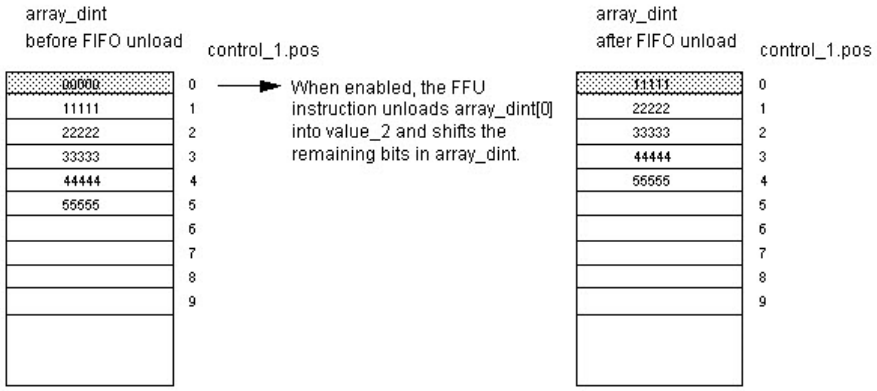

Rung-condition-in is true
.LEN <= 0
Yes
No
.POS < 0
Yes
No
Set .DN to true
Set .EM to true
Clear .DN to false
Clear .EM to false
.EU is true
No
Yes
Set .EU to true
.POS = .POS + 1
.POS <= 1
Yes
Set .EM to true
No
.POS < 1
No
Yes
Destination = 0
.POS = 0
Yes
Set .EM to true
No
.POS >= .LEN
No
Yes
Set .DN to true
.POS > .LEN
Yes
.POS = .LEN
No
.POS = .POS - 1
FIFO[.LEN] is past the end of the array
Yes
Major Fault
No
Destination = FIFO[0]
Shift the FIFO down 1 element (FIFO[x] = FIFO[x+1])
FIFO[.LEN – 1] = 0
End

예

예 1

래더 다이어그램

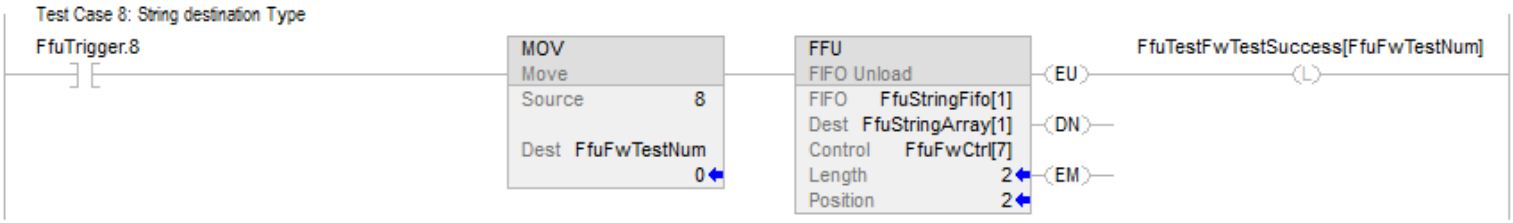




예 2

대상 배열이 STRING 배열이거나 구조 배열입니다

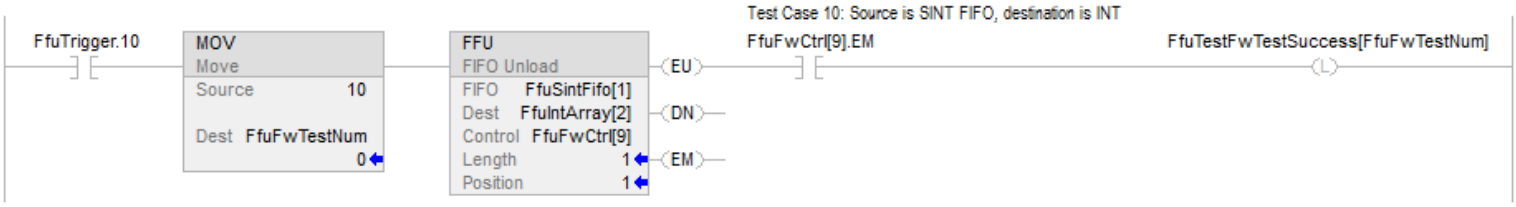
래더 다이어그램



예 3

FIFO 소스 배열의 데이터 유형이 대상 배열의 데이터 유형과 일치하지 않습니다.

래더 다이어그램



추가 참조

배열(파일)/이동 명령어 페이지의 619

공통 속성 페이지의 963

FFL 페이지의 630

LFL 페이지의 646

LFU 페이지의 654

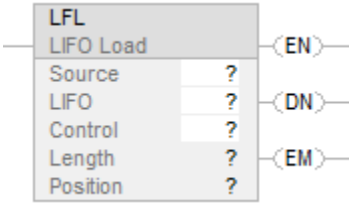
LIFO 로드(LFL)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

LFL 명령어는 Source 값을 LIFO 에 복사합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT REAL 문자열 유형 구조	즉시 태그	LIFO 에 저장할 데이터.
LIFO	SINT INT DINT REAL 문자열 유형 구조	배열 태그	수정할 LIFO. LIFO 의 첫 번째 요소를 지정합니다.
Control	CONTROL	태그	연산의 제어 구조 일반적으로 연결된 LFU 와 동일한 CONTROL 을 사용합니다.
Length	DINT	즉시	LIFO 가 한 번에 수용할 수 있는 최대 요소 수.
Position	DINT	즉시	명령어가 데이터를 로드할 LIFO 의 다음 위치 초기값은 일반적으로 0 입니다.

CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 LFL 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 LIFO 가 가득 찼음을(.POS = .LEN) 나타내도록 설정됩니다. .DN 비트가 .POS < .LEN 이 될 때까지 LIFO 를 로드하지 못하게 합니다.
.EM	BOOL	빈 비트는 LIFO 가 비어 있음을 나타냅니다. .LEN < 0, .LEN = 0 또는 .POS < 0 이면 .EM 비트와 .DN 비트가 설정됩니다.
.LEN	DINT	길이는 LIFO 가 한 번에 수용할 수 있는 최대 요소 수를 지정합니다.
.POS	DINT	위치는 명령어가 다음 값을 로드할 LIFO 의 위치를 식별합니다.

설명

LFL 명령어와 LFU 명령어를 함께 사용하여 후입선출의 순서로 데이터를 저장하고 검색합니다. LFL 명령어와 LFU 명령어를 한 쌍으로 사용할 경우 비동기식 이동 레지스터가 구성됩니다.

일반적으로 Source 와 LIFO 의 데이터 유형이 동일합니다.

활성화된 경우 LFL 명령어는 LIFO 에서 .POS 값으로 식별한 위치에 소스 값을 로드합니다. 이 명령어는 LIFO 가 가득 찰 때까지 활성화될 때마다 값을 한 개씩 로드합니다.

중요:	명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

LFL 명령어는 연속 데이터 메모리에서 작동됩니다. CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러의 경우 명령어의 범위가 기본 태그로 제한됩니다.
일반적으로 Source 와 LIFO 의 데이터 유형이 동일합니다. Source 와 LIFO 의 데이터 유형이 일치하지 않을 경우 이 명령어가 Source

값을 FIFO 태그의 데이터 유형으로 변환합니다. 작은 정수는 부호 확장을 통해 큰 정수로 변환됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
(시작 요소 + .POS)가 LIFO 배열의 끝을 지난 경우	4	20

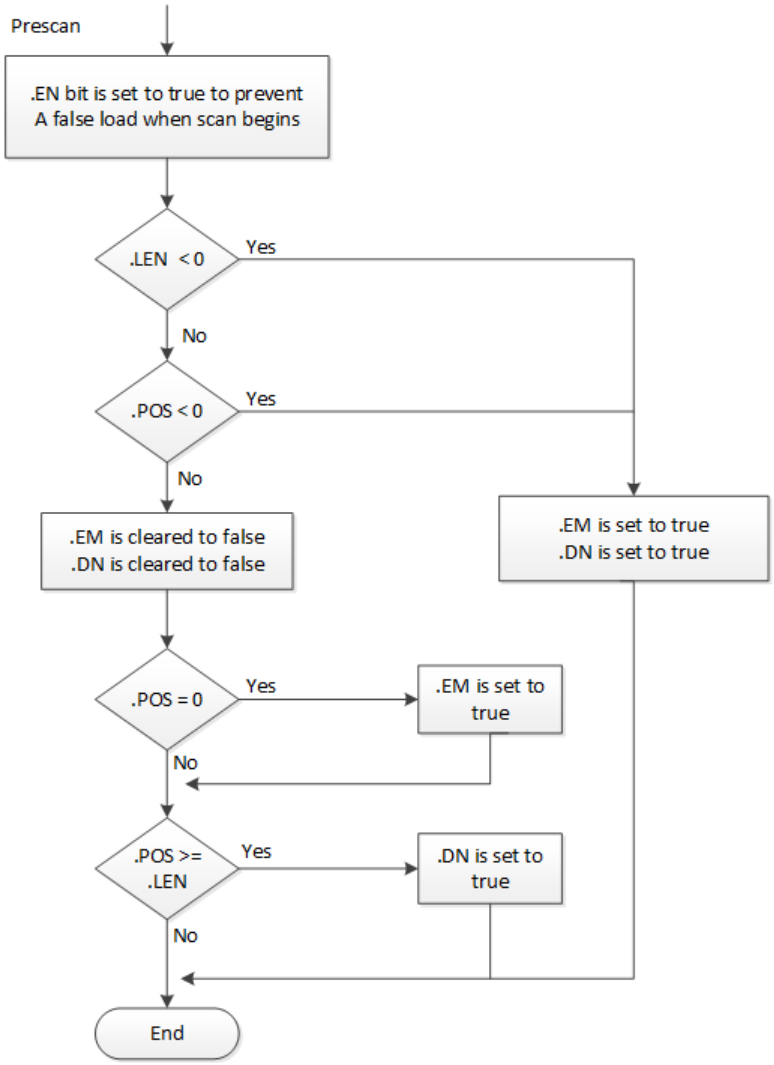
피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

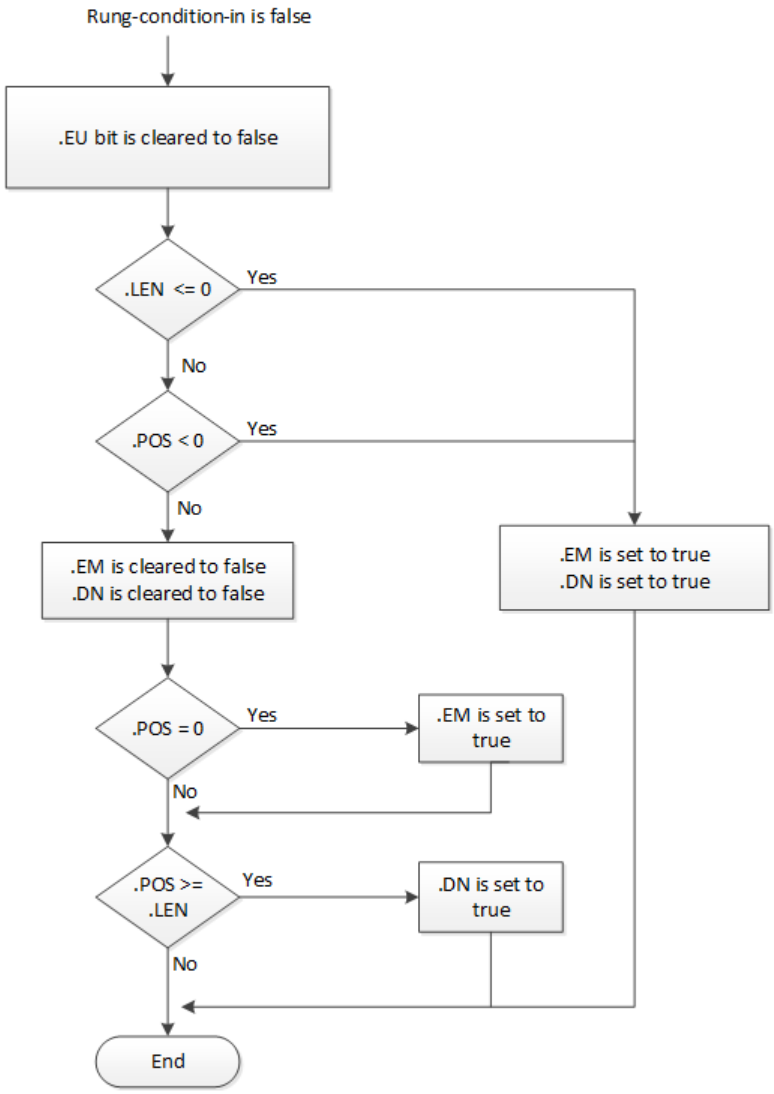
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	LFL 흐름도(사전 스캔)를 참조하십시오.
렁-입력-조건이 거짓임	LFL 흐름도(거짓)를 참조하십시오.
렁-입력-조건이 참임	LFL 흐름도(참)를 참조하십시오.
사후 스캔	N/A.

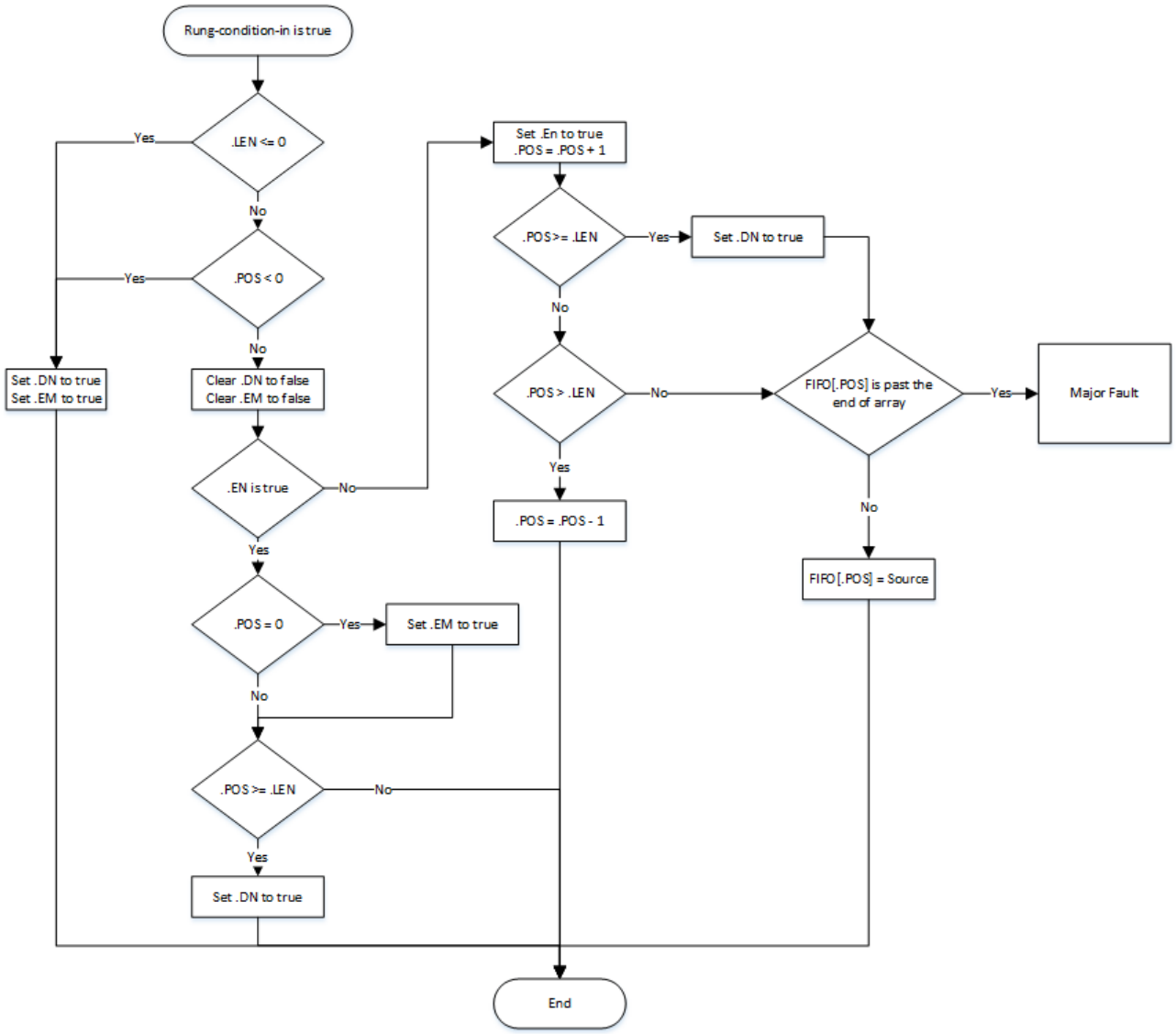
LFL 흐름도(사전 스캔)


Prescan
.EN bit is set to true to prevent A false load when scan begins
.LEN < 0
Yes
No
.POS < 0
Yes
No
.EM is cleared to false
.DN is cleared to false
.EM is set to true
.DN is set to true
.POS = 0
Yes
.EM is set to true
No
.POS >= .LEN
Yes
.DN is set to true
No
End

LFL 흐름도(거짓)


Rung-condition-in is false
.EU bit is cleared to false
.LEN <= 0
Yes
No
.POS < 0
Yes
No
.EM is cleared to false
.DN is cleared to false
.EM is set to true
.DN is set to true
.POS = 0
Yes
.EM is set to true
No
.POS >= .LEN
Yes
.DN is set to true
No
End

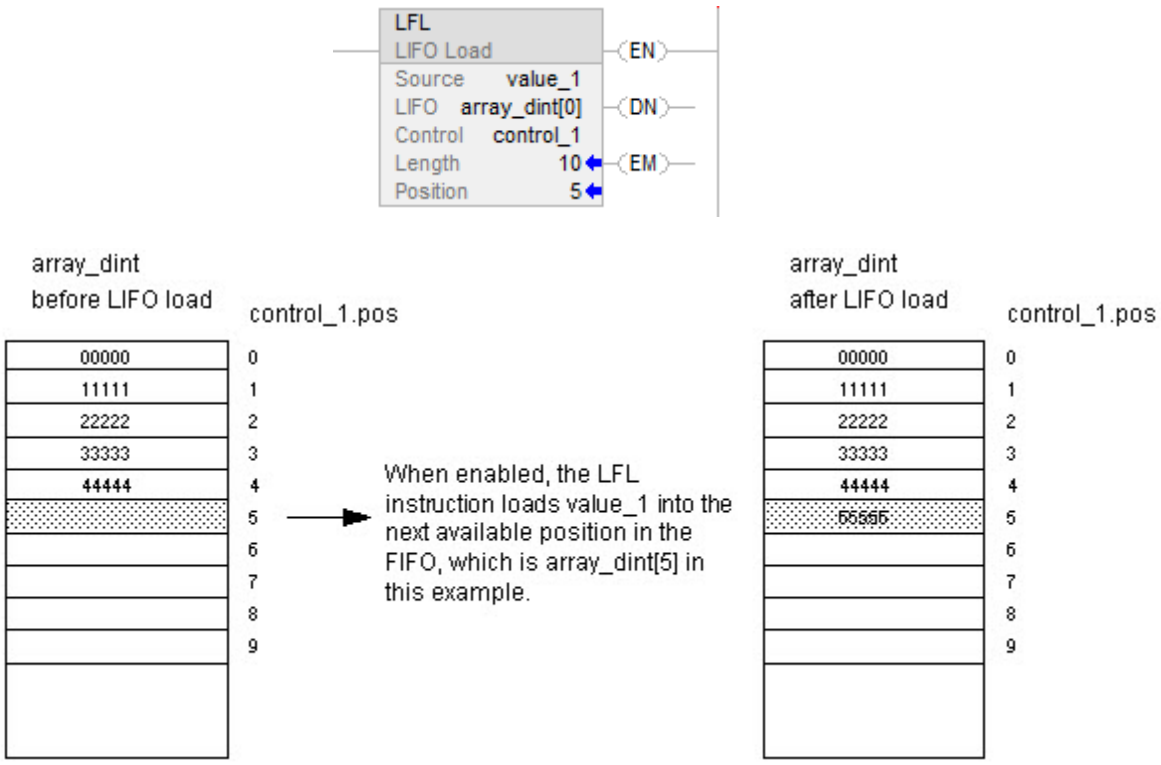
LFL 흐름도(참)


Rung-condition-in is true
.LEN <= 0
Yes
No
.POS < 0
Yes
No
Set .DN to true
Set .EM to true
Clear .DN to false
Clear .EM to false
.EN is true
No
Yes
.POS = 0
Yes
Set .EM to true
No
.POS >= .LEN
No
Yes
Set .DN to true
Set .En to true
.POS = .POS + 1
.POS>= .LEN
Yes
Set .DN to true
No
.POS > .LEN
No
FIFO[.POS] is past the end of array
Yes
Major Fault
Yes
.POS = .POS - 1
No
FIFO [.POS] = Source
End

예

예 1

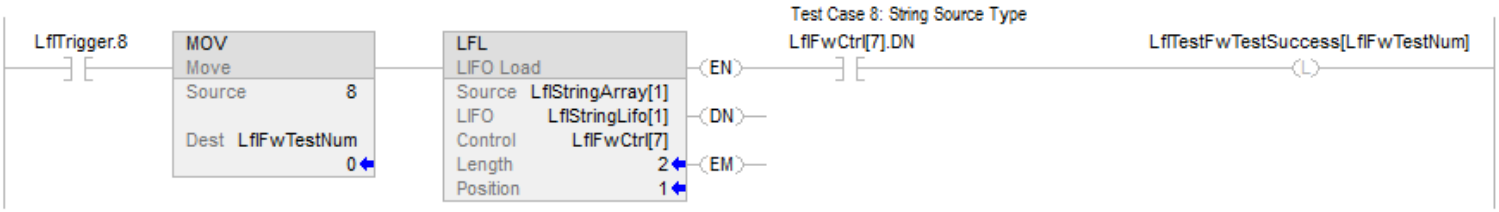
래더 다이어그램



예 2

Source 배열이 STRING 배열이거나 구조 배열입니다.

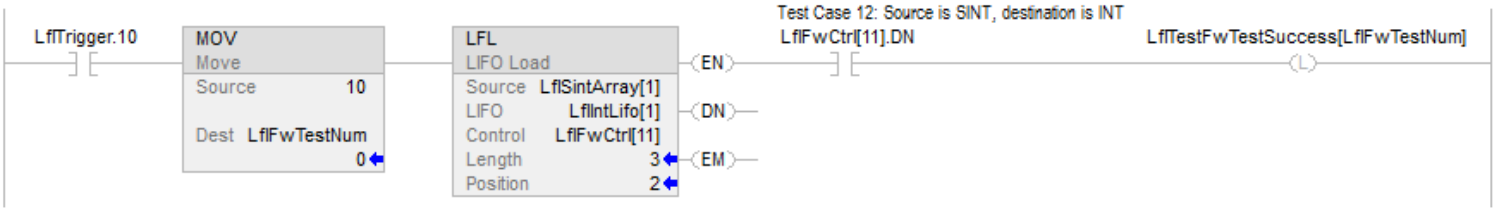
래더 다이어그램



예 3

Source 의 데이터 유형이 LIFO 배열의 데이터 유형과 일치하지 않습니다.

래더 다이어그램



추가 참조

배열(파일)/이동 명령어 페이지의 619

LIFO 언로드(LFU) 페이지의 654

FIFO 로드(FFL) 페이지의 630

FIFO 언로드(FFU) 페이지의 638

공통 속성 페이지의 963

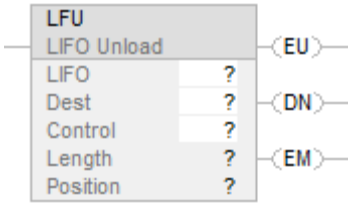
LIFO 언로드(LFU)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

LFU 명령어는 LIFO 의 .POS 의 값을 언로드하고 해당 위치에 0 을 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다.

래더 다이어그램

피연산자	유형	형식	설명
LIFO	SINT INT DINT REAL 문자열 유형 구조	배열 태그	수정할 LIFO. LIFO 의 첫 번째 요소를 지정합니다. CONTROL.POS 를 첨자로 사용하지 마십시오.
Destination	SINT INT DINT REAL 문자열 유형 구조	태그	LIFO 에서 언로드된 값.
Control	CONTROL	태그	연산의 제어 구조 일반적으로 연결된 LFL 과 동일한 CONTROL 을 사용합니다.
Length	DINT	즉시	LIFO 가 한 번에 수용할 수 있는 최대 요소 수.
Position	DINT	즉시	명령어가 데이터를 언로드할 LIFO 의 다음 위치. 초기값은 일반적으로 0 입니다.

CONTROL 구조

니모닉	데이터 유형	설명
.EU	BOOL	활성화 비트는 LFU 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 LIFO 가 가득 찼음을(.POS = .LEN) 나타내도록 설정됩니다.
.EM	BOOL	빈 비트는 LIFO 가 비어 있음을 나타냅니다. .LEN < 0, .LEN = 0 또는 .POS < 0 이면 .EM 비트와 .DN 비트가 모두 설정됩니다.
.LEN	DINT	길이는 LIFO 가 한 번에 수용할 수 있는 최대 요소 수를 지정합니다.
.POS	DINT	위치는 LIFO 에 로드된 데이터의 끝을 식별합니다.

설명

LFU 명령어와 LFL 명령어를 함께 사용하여 후입선출의 순서로 데이터를 저장하고 검색합니다.

활성화된 경우 LFU 명령어는 LIFO 의 .POS 에 값을 언로드하고 해당 값을 Destination 에 저장합니다. 이 명령어는 LIFO 가 빌 때까지 활성화될 때마다 값을 한 개씩 언로드하고 해당 값을 0 으로 바꿉니다. LIFO 가 비면 LFU 가 Destination 에 0 을 반환합니다.

중요: 명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

LFU 명령어는 연속 메모리에서 작동됩니다. 명령어의 범위가 기본 태그로 제한됩니다. LFL 명령어는 기본 태그의 외부에 데이터를 쓰지 않지만 구성원 경계를 넘을 수 있습니다. 구조의 구성원인 배열을 지정하고 길이가 해당 배열의 크기를 초과할 경우 LFL 명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에서는 데이터가 지정된 구성원으로 제한됩니다.

이 명령어가 배열의 끝을 지나서 읽기 작업을 수행하려고 할 경우 .ER 비트를 설정하여 메이저 폴트가 발생합니다.

일반적으로 Source 와 LIFO 의 데이터 유형이 동일합니다. Source 와 LIFO 의 데이터 유형이 일치하지 않을 경우 이 명령어가 Source 값을 FIFO 태그의 데이터 유형으로 변환합니다.

작은 정수는 부호 확장을 통해 큰 정수로 변환됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
지정된 Length 가 LIFO 배열의 끝을 지난 경우	4	20

피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

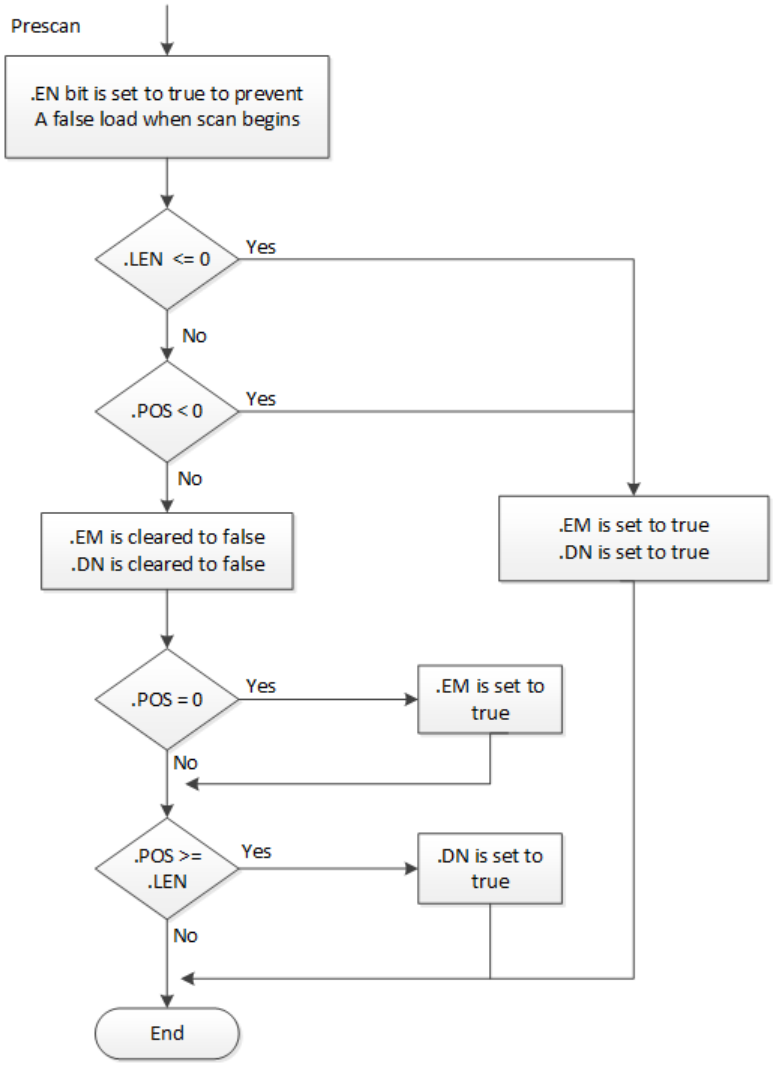
실행

모든 조건이 정상 스캔 모드에서만 발생합니다.

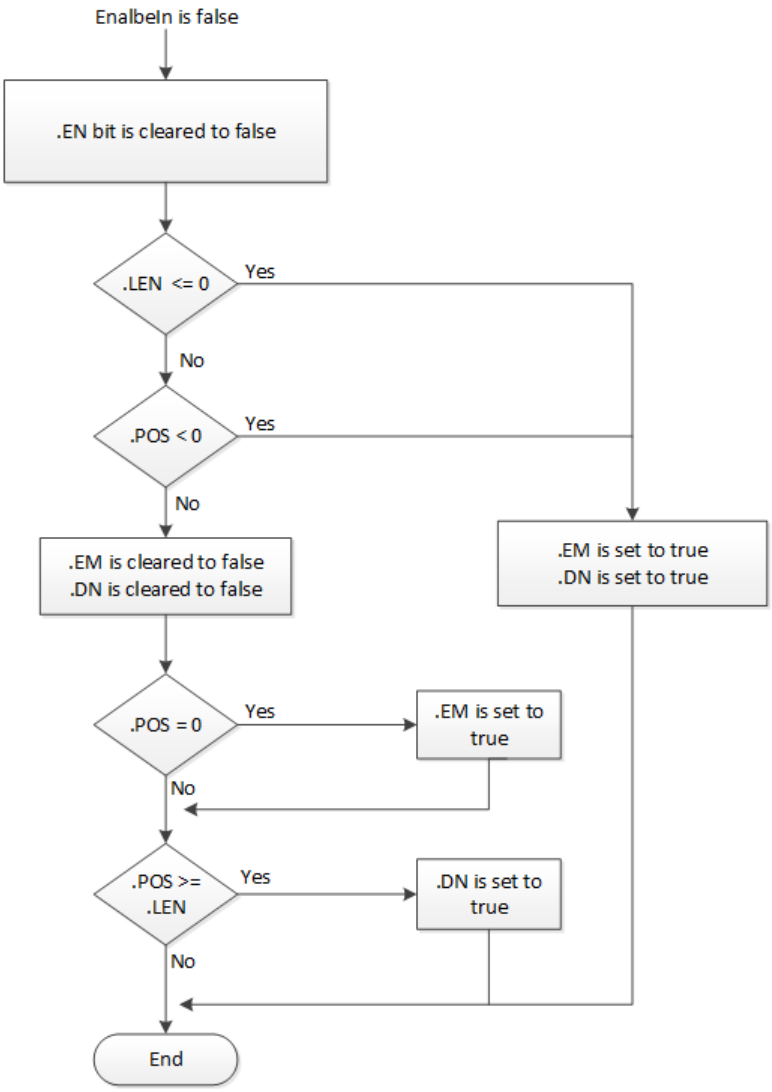
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	LFU 흐름도(사전 스캔)를 참조하십시오.
렁-입력-조건이 거짓임	LFU 흐름도(거짓)를 참조하십시오.
렁-입력-조건이 참임	LFU 흐름도(참)를 참조하십시오.
사후 스캔	N/A

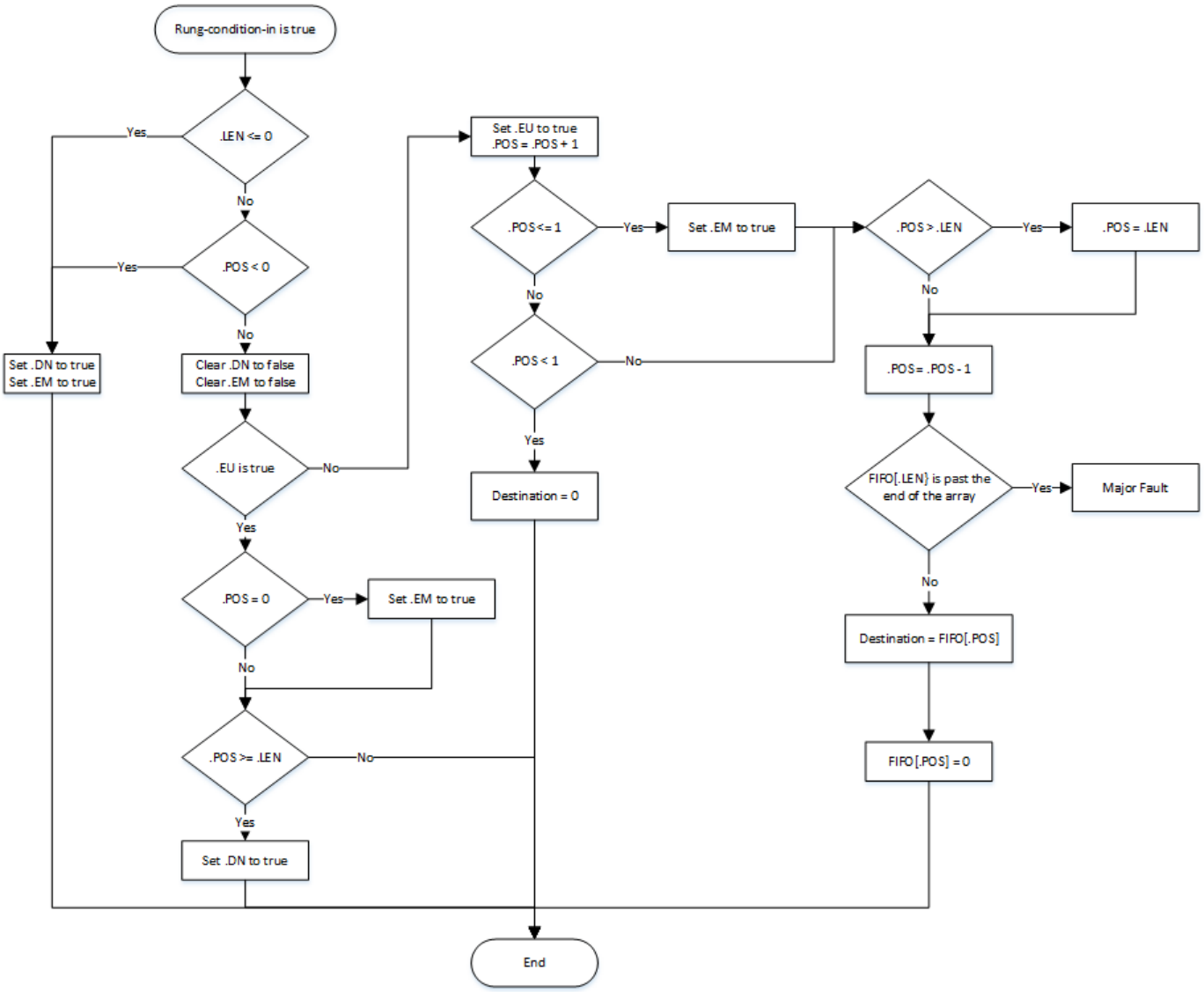
LFU 흐름도(사전 스캔)


Prescan
.EN bit is set to true to prevent
A false load when scan begins
.LEN <= 0
Yes
No
.POS < 0
Yes
No
.EM is cleared to false
.DN is cleared to false
.EM is set to true
.DN is set to true
.POS = 0
Yes
.EM is set to true
No
.POS >= .LEN
Yes
.DN is set to true
No
End

LFU 흐름도(거짓)


EnalbeIn is false
.EN bit is cleared to false
.LEN <= 0
Yes
No
.POS < 0
Yes
No
.EM is cleared to false
.DN is cleared to false
.EM is set to true
.DN is set to true
.POS = 0
Yes
.EM is set to true
No
.POS >= .LEN
Yes
.DN is set to true
No
End

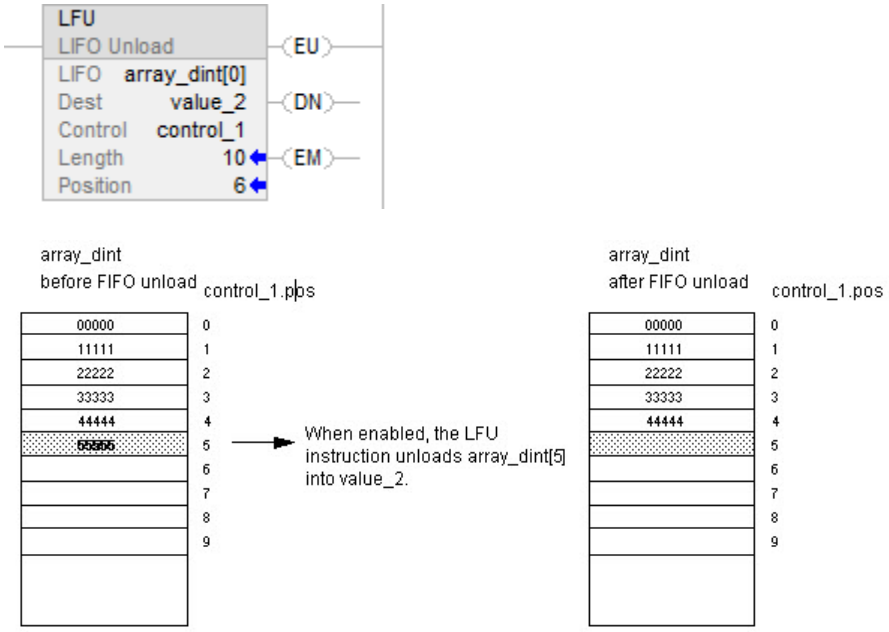
LFU 흐름도(참)


Rung-condition-in is true
.LEN <= 0
Yes
No
.POS < 0
Yes
No
Set .DN to true
Set .EM to true
Clear .DN to false
Clear .EM to false
.EU is true
No
Yes
.POS = 0
Yes
Set .EM to true
No
.POS >= .LEN
No
Yes
Set .DN to true
Set .EU to true
.POS = .POS + 1
.POS <= 1
Yes
Set .EM to true
No
.POS < 1
No
Yes
Destination = 0
.POS > .LEN
Yes
.POS = .LEN
No
.POS = .POS - 1
FIFO[.LEN} is past the end of the array
Yes
Major Fault
No
Destination = FIFO[.POS]
FIFO[.POS] = 0
End

예

예 1

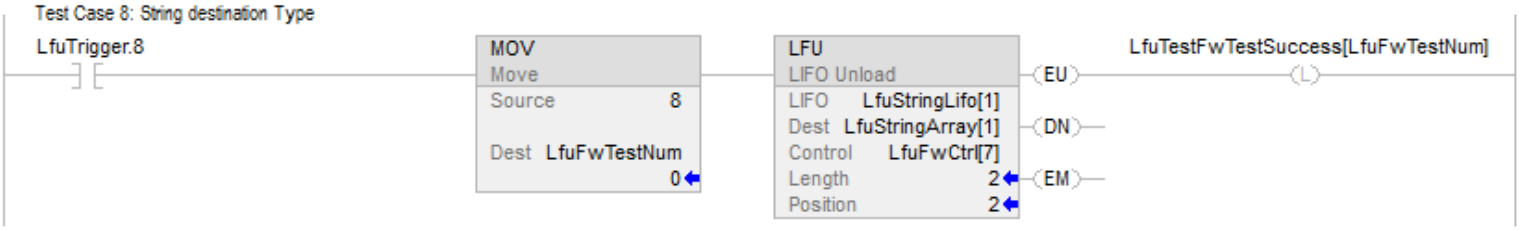
래더 다이어그램



예 2

대상 배열이 STRING 배열이거나 구조 배열입니다

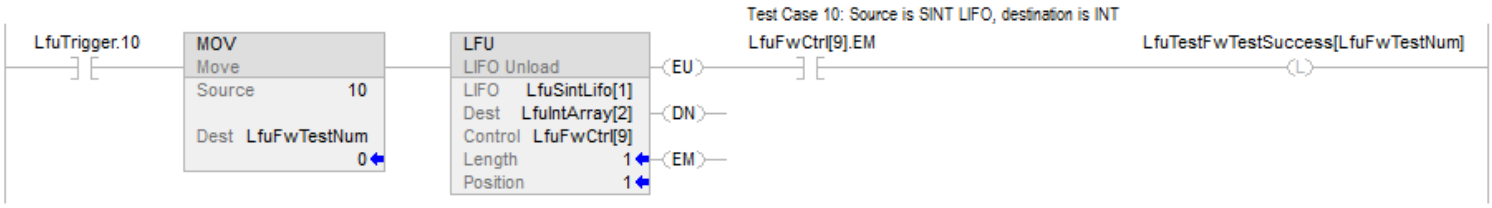
래더 다이어그램



예 3

LIFO 소스 배열의 데이터 유형이 대상 배열의 데이터 유형과 일치하지 않습니다.

래더 다이어그램



추가 참조

배열(파일)/이동 명령어 페이지의 619

LIFO 로드(LFL) 페이지의 646

FIFO 로드(FFL) 페이지의 630

FIFO 언로드(FFU) 페이지의 638

공통 속성 페이지의 963

시퀀서 명령어

시퀀서 명령어

시퀀서 명령어는 일관되고 반복 가능한 작업을 모니터링합니다.

사용 가능한 명령어

래더 다이어그램

SQI	SQO	SQL

평션 블록

사용할 수 없음

ST(스트럭처드 텍스트)

사용할 수 없음

실행할 작업	사용할 명령어
단계 완료를 감지합니다.	SQI
다음 단계의 출력 조건을 설정합니다.	SQO
참조 조건을 시퀀서 배열에 로드합니다.	SQL

볼드체 데이터 유형은 최적의 데이터 유형을 나타냅니다. 명령어의 모든 피연산자가 동일한 최적의 데이터 유형(일반적으로 DINT 또는 REAL)을 사용하는 경우 명령어는 가장 적은 메모리를 사용하면서 보다 빠르게 실행됩니다.

추가 참조

계산/연산 명령어 페이지의 411

비교 명령어 페이지의 329

비트 명령어 페이지의 81

ASCII 문자열 명령어 페이지의 907

ASCII 변환 명령어 페이지의 927

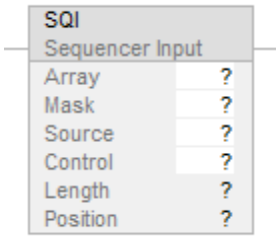
시퀀서 입력(SQI)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

SQI 명령어는 SQO/SQI 명령어의 시퀀스 쌍에서 단계가 완료되는 시점을 감지합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형의 데이터 변환 규칙. 데이터 변환을 참조하십시오.

피연산자	유형	형식	설명
Array	DINT	배열 태그	시퀀서 배열 시퀀서 배열의 첫 번째 요소를 지정합니다. 첨자에 CONTROL.POS 를 사용하지 마십시오.
Mask	SINT INT DINT	태그 즉시	이 피연산자는 Source 와.POS 에서 참조되는 배열에 적용된 경우 차단하거나(0) 통과시킬(1) 비트를 결정하는 데 사용됩니다. INT 및 SINT 유형이 0 으로 DINT 유형의 크기로 확장됩니다.
Source	SINT INT DINT	태그 즉시	.POS 에서 참조되는 배열 요소와 비교하는 데 사용되는 입력 데이터.
Control	CONTROL	태그	연산의 제어 구조 SQO 명령어와 SQL 명령어에 동일한 제어 태그를 사용해야 합니다.
Length	DINT	즉시	CONTROL 구조 .LEN 을 나타냅니다.
Position	DINT	즉시	CONTROL 구조 .POS 를 나타냅니다.

CONTROL 구조

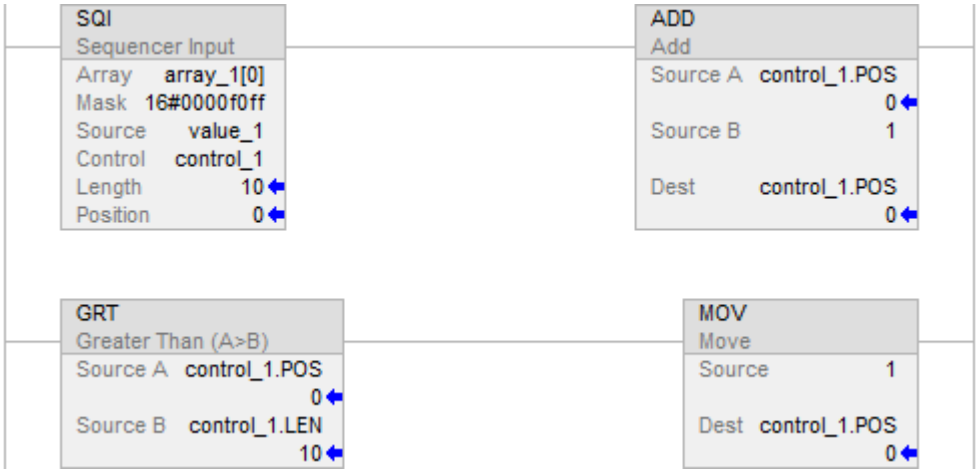
니모닉	데이터 유형	설명
.ER(에러)	BOOL	명령어에서 에러가 발생했습니다.
.LEN(길이)	DINT	길이는 시퀀서 배열의 시퀀서 단계 수를 지정합니다.
.POS(위치)	DINT	위치는 명령어가 현재 Source 와 비교하는 Array 요소를 식별합니다. 초기값은 일반적으로 0 입니다.

설명

참인 경우 SQI 명령어가 Mask 를 통해 Source 와 현재 Array 요소를 통과시킵니다. 이러한 마스킹 연산의 결과가 비교되며, 비교 결과가 동일한 경우 렁-출력-조건이 참으로 설정되고, 다른 경우 렁-출력-조건이 거짓으로 해제됩니다. 일반적으로 SQO 및 SQL 명령어와 동일한 CONTROL 구조를 사용합니다.

SQO 없이 SQI 사용

SQI 명령어가 단계가 완료되었다고 결정하면 ADD 명령어가 시퀀서 배열을 증분합니다. GRT 가 시퀀서 배열 내에서 다른 값을 검사할 수 있는지 결정합니다. MOV 명령어가 시퀀서 배열을 단계별로 완전히 실행한 후 위치 값을 리셋합니다.



연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

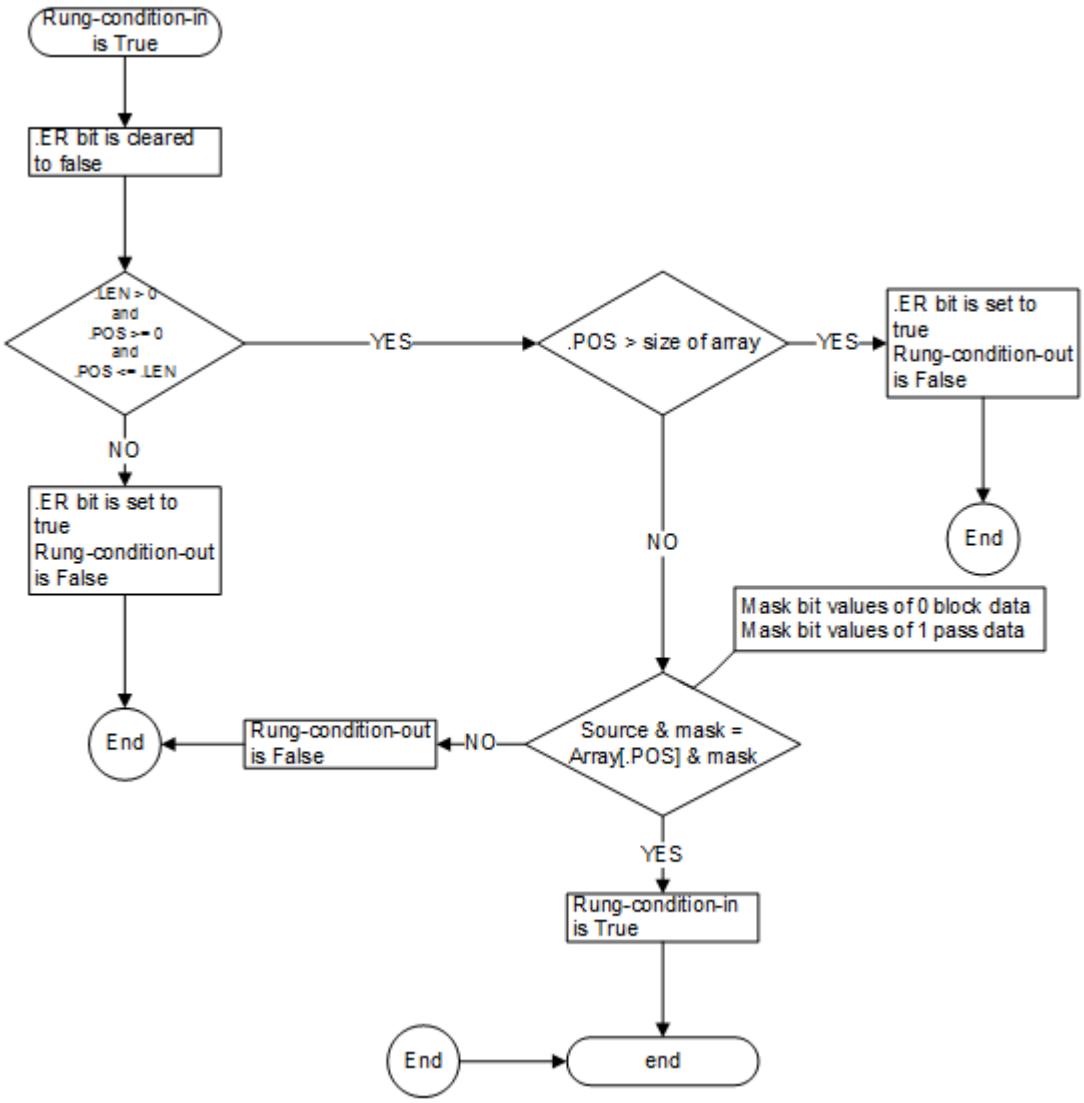
이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

래더 다이어그램

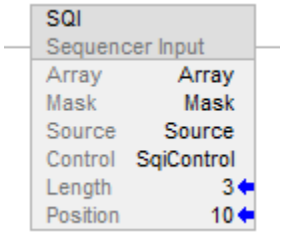
조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	흐름도(참) 참조하십시오.
사후 스캔	N/A

흐름도(참)



예

래더 다이어그램



조합된 SQO 명령어 없이 SQI 명령어를 사용할 경우 외부에서 시퀀서 배열을 증가해야 합니다.

enableOut 명령어가 참일 때 Position(예: 마스크 값이 있는 Array[Position])에 지정된 배열 값과 Mask 값의 논리곱 연산의 결과가 소스 값과 Mask 값의 논리곱 연산의 결과와 동일한 경우 렁-입력-조건이 참으로 설정되고, 그렇지 않을 경우 렁-출력-조건이 거짓으로 해제됩니다.

추가 참조

시퀀서 명령어 페이지의 663

공통 속성 페이지의 963

데이터 변환 페이지의 967

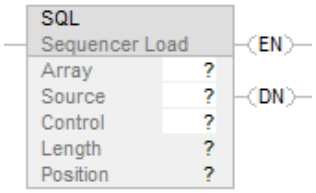
시퀀서 로드(SQL)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

SQL 명령어는 소스 피연산자 값을 시퀀서 배열에 로드합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형의 데이터 변환 규칙. 데이터 변환을 참조하십시오.

피연산자	유형	형식	설명
배열	DINT	배열 태그	시퀀서 배열 시퀀서 배열의 첫 번째 요소를 지정합니다. 첨자에 CONTROL.POS 를 사용하지 마십시오.
소스(Source)	SINT INT DINT	태그 즉시	.POS 에 지정된 위치에서 시퀀서 어레이에 로드할 데이터
제어(Control)	CONTROL	태그	연산의 제어 구조 SQI 명령어와 SQO 명령어에 동일한 제어 태그를 사용해야 합니다.
길이(Length)	DINT	즉시	CONTROL 구조 .LEN 을 나타냅니다.
위치(Position)	DINT	즉시	CONTROL 구조 .POS 를 나타냅니다.

CONTROL 구조

니모닉	데이터 유형	설명
.EN(활성화)	BOOL	활성화 비트는 SQL 명령어가 활성화되었음을 나타냅니다.
.DN(완료)	BOOL	완료 비트는 지정된 모든 요소가 배열에 로드된 경우에 설정됩니다.

니모닉	데이터 유형	설명
.ER(에러)	BOOL	에러 비트는 .LEN < 0, .LEN = 0, .POS < 0 또는 .POS > .LEN 인 경우에 설정됩니다.
.LEN(길이)	DINT	길이는 시퀀서 배열의 시퀀서 단계 수를 지정합니다.
.POS(위치)	DINT	위치는 배열에서 Source 값이 저장되는 위치를 식별합니다.

설명

.EN 이 거짓에서 참으로 바뀌면 .POS 가 증가됩니다. .POS 는 .POS > 또는 = .LEN 인 경우 1 로 리셋됩니다. SQL 명령어가 배열의 새 위치에 Source 값을 로드합니다.

.EN 이 참이면 SQL 명령어가 배열의 현재 위치에 Source 값을 로드합니다.

일반적으로 SQI 및 SQO 명령어와 동일한 CONTROL 구조를 사용합니다.

중요: 명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

연산 상태 플래그에 영향

아니요

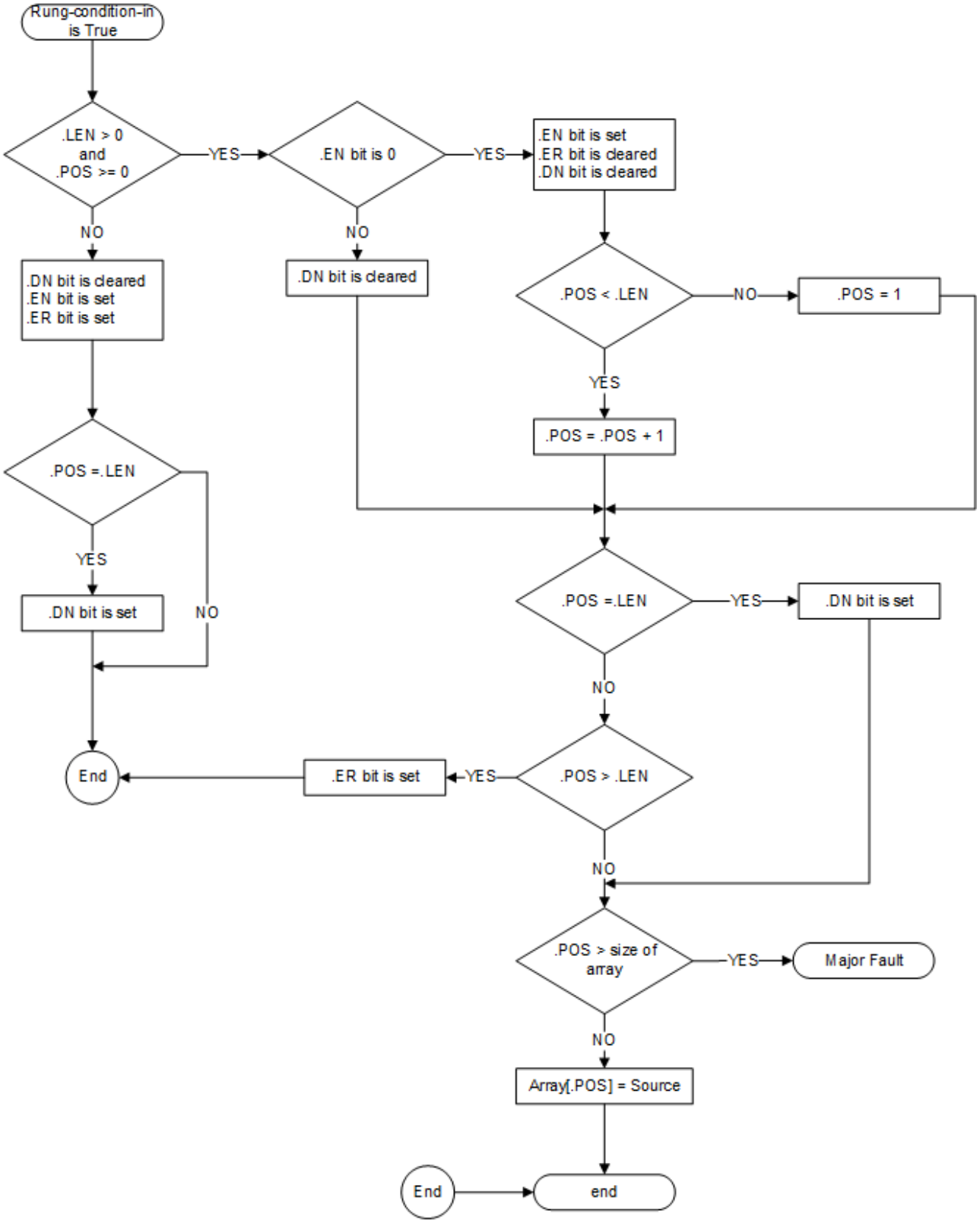
메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
위치 > 배열 크기	4	20

실행

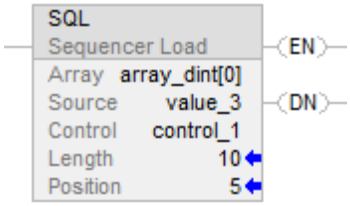
조건/상태	취해진 조치
사전 스캔	.EN 이 참으로 설정됩니다.
렁-입력-조건이 거짓임	.EN 이 거짓으로 해제됩니다.
렁-입력-조건이 참임	흐름도(참) 참조하십시오.
사후 스캔	N/A

흐름도 - 참


Rung-condition-in is True
.LEN > 0 and .POS >= 0
YES
.EN bit is 0
YES
.EN bit is set
.ER bit is cleared
.DN bit is cleared
NO
.DN bit is cleared
.EN bit is set
.ER bit is set
NO
.DN bit is cleared
.POS < .LEN
NO
.POS = 1
YES
.POS = .POS + 1
.POS =.LEN
YES
.DN bit is set
NO
.POS =.LEN
YES
.DN bit is set
NO
End
.ER bit is set
YES
.POS > .LEN
NO
.POS > size of array
YES
Major Fault
NO
Array[.POS] = Source
End
end

예

래더 다이어그램



활성화된 경우 SQL 명령어는 value_3 을 시퀀서 배열의 다음 위치에 로드합니다(이 예에서는 array_dint[5]).

추가 참조

시퀀서 명령어 페이지의 663

SQO 페이지의 672

SQI 페이지의 664

공통 속성 페이지의 963

데이터 변환 페이지의 967

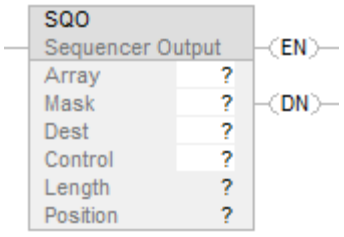
시퀀서 출력(SQO)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

SQO 명령어는 SQO/SQI 명령어 시퀀스 쌍의 다음 단계에 대한 출력 조건을 설정합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형의 데이터 변환 규칙. 데이터 변환을 참조하십시오.

피연산자	유형	형식	설명
Array	DINT	배열 태그	시퀀서 배열 시퀀서 배열의 첫 번째 요소를 지정합니다. 첨자에 CONTROL.POS 를 사용하지 마십시오.
Mask	SINT INT DINT	태그 즉시	차단(0) 또는 전달할(1) 비트를 결정하기 위해 사용되며 출력 마스킹 연산 중에 적용됩니다.
Destination	DINT	태그	시퀀서 배열에서의 출력 데이터. 이 값은 출력 마스킹 연산에 사용됩니다.
Control	CONTROL	태그	연산의 제어 구조 SQI 명령어와 SQL 명령어에 동일한 제어 태그를 사용해야 합니다.
Length	DINT	즉시	출력에 대한 배열(시퀀서 표)의 요소 수
Position	DINT	즉시	배열에서의 현재 위치 초기값은 일반적으로 0 입니다.

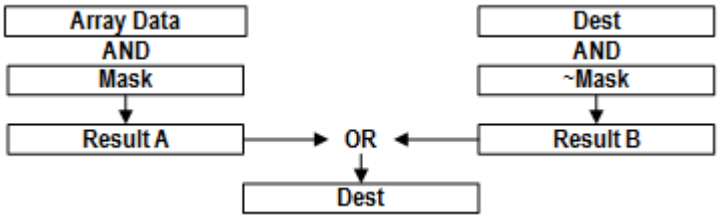
CONTROL 구조

니모닉	데이터 유형	설명
.EN(활성화)	BOOL	활성화 비트는 SQO 명령어가 활성화되었음을 나타냅니다.
.DN(완료)	BOOL	완료 비트는 .POS = .LEN 인 경우에 설정됩니다.
.ER(에러)	BOOL	명령어에서 에러가 발생했음을 나타냅니다.
.LEN(길이)	DINT	길이는 시퀀서 배열의 시퀀서 단계 수를 지정합니다.
.POS(위치)	DINT	위치는 명령어가 현재 출력 마스킹 연산에 사용 중인 배열 요소를 식별합니다.

설명

.EN 이 거짓에서 참으로 바뀌면 .POS 가 증가됩니다. .POS 는 .POS 가 .LEN 이상이 될 때 1 로 리셋됩니다.

.EN 이 참이면 SQO 명령어는 Mask 를 통해.POS 에 있는 배열 데이터를 이동한 다음 Mask 의 보수를 통해 현재 Destination 값을 이동합니다. 이러한 연산의 결과를 함께 논리합 연산하고 결과를 Destination 에 저장합니다.



일반적으로 SQI 및 SQL 명령어와 같은 CONTROL 구조를 사용해야 합니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

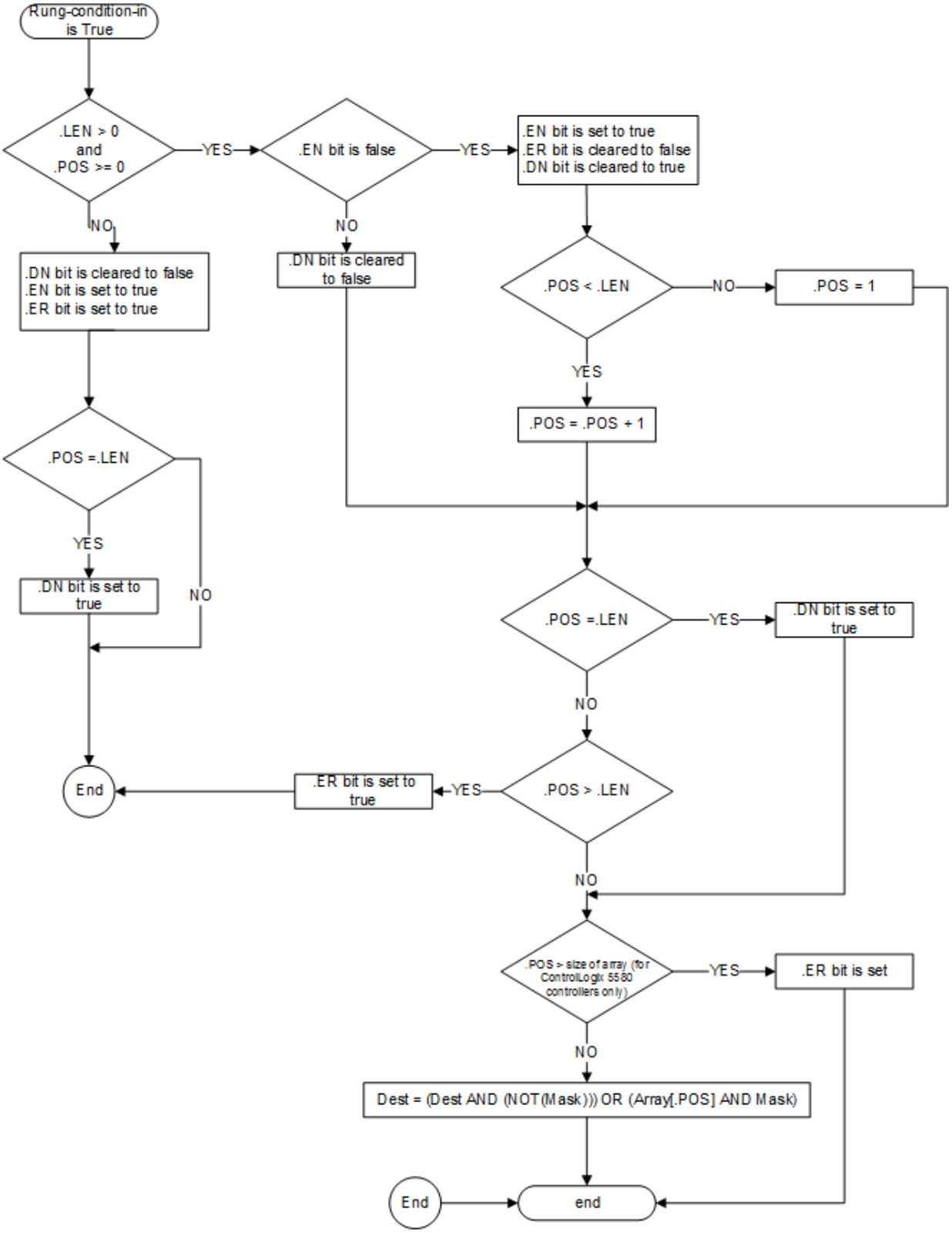
이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	.EN 이 참으로 설정됩니다.
렁-입력-조건이 거짓임	.EN 이 거짓으로 해제됩니다.
렁-입력-조건이 참임	다음 흐름도(참) 참조하십시오.
사후 스캔	N/A

흐름도(참)

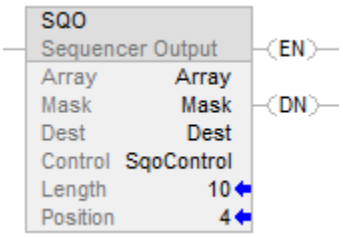

Rung-condition-in is True
.LEN > 0 and .POS >= 0
YES
.EN bit is false
YES
.EN bit is set to true
.ER bit is cleared to false
.DN bit is cleared to true
NO
.DN bit is cleared to false
.EN bit is set to true
.ER bit is set to true
NO
.DN bit is cleared to false
.POS < .LEN
NO
.POS = 1
YES
.POS = .POS + 1
.POS =.LEN
YES
.DN bit is set to true
NO
.POS =.LEN
YES
.DN bit is set to true
NO
End
.ER bit is set to true
YES
.POS > .LEN
NO
.POS > size of array (for ControlLogix 5580 controllers only)
YES
.ER bit is set
NO
Dest = (Dest AND (NOT(Mask))) OR (Array[.POS] AND Mask)
End
end

예

Mask 값은 배열 값과 논리곱 연산됩니다. 예: Array[SqoControl.POS]. Mask 값의 보수는 현재 Dest 값과 논리곱 연산됩니다. 이러한 두 연산의 결과를 함께 논리합 연산하고 결과를 Dest 에 저장합니다.

.POS 를 초기 값(.POS = 0)으로 리셋하려면 RES 명령어를 사용하여 제어 구조를 해제합니다. 이 예는 첫 번째 스캔 비트의 상태를 사용하여 .POS 값을 해제합니다.

래더 다이어그램



추가 참조

시퀀서 명령어 페이지의 663

SQI 페이지의 664

SQL 페이지의 668

공통 속성 페이지의 963

데이터 변환 페이지의 967

프로그램 제어 명령어

프로그램 제어 명령어를 사용하여 로직 흐름을 변경하십시오.

사용 가능한 명령어

래더 다이어그램

JMP	LBL	JSR	JXR	RET	SBR	TND	MCR

UID	UIE	SFR	SFP	EVENT	AFI	EOT	NOP

평션 블록

JSR	RET	SBR

ST(스트럭처드 텍스트)

JSR	RET	SBR	TND	EVENT	UID	EOT	SFR

UIE	SFP

실행할 작업:	사용할 명령어:
항상 실행될 필요가 없는 로직 섹션으로 이동합니다.	JMP LBL
별도의 루틴으로 이동하고, 루틴에 데이터를 전달하며, 루틴을 실행하고, 결과를 반환합니다.	JSR SBR RET

외부 루틴으로 이동합니다.	JXR
루틴 실행을 중단시키는 임시 종료를 표시합니다.	TND
로직 섹션에서 모든 렁을 비활성화합니다.	MCR
사용자 태스크를 비활성화합니다.	UID
사용자 태스크를 활성화합니다.	UIE
순차 펑션 차트를 일시 중지합니다.	SFP
순차 펑션 차트를 리셋합니다.	SFR
순차 펑션 차트의 전환을 종료합니다.	EOT
이벤트 태스크의 실행을 트리거합니다.	EVENT
렁을 비활성화합니다.	AFI
로직에 자리 표시자를 삽입합니다.	NOP

추가 참조

계산/연산 명령어 페이지의 411

비교 명령어 페이지의 329

비트 명령어 페이지의 81

ASCII 문자열 명령어 페이지의 907

ASCII 변환 명령어 페이지의 927

프로그램 제어 명령어

프로그램 제어 명령어를 사용하여 로직 흐름을 변경하십시오.

사용 가능한 명령어

래더 다이어그램

JMP	LBL	JSR	JXR	RET	SBR	TND	MCR

UID	UIE	SFR	SFP	EVENT	AFI	EOT	NOP

펑션 블록

JSR	RET	SBR

ST(스트럭처드 텍스트)

JSR	RET	SBR	TND	EVENT	UID	EOT	SFR

UIE	SFP

실행할 작업:	사용할 명령어:
항상 실행될 필요가 없는 로직 섹션으로 이동합니다.	JMP LBL
별도의 루틴으로 이동하고, 루틴에 데이터를 전달하며, 루틴을 실행하고, 결과를 반환합니다.	JSR SBR RET
외부 루틴으로 이동합니다.	JXR
루틴 실행을 중단시키는 임시 종료를 표시합니다.	TND
로직 섹션에서 모든 렁을 비활성화합니다.	MCR
사용자 태스크를 비활성화합니다.	UID
사용자 태스크를 활성화합니다.	UIE
순차 펑션 차트를 일시 중지합니다.	SFP
순차 펑션 차트를 리셋합니다.	SFR
순차 펑션 차트의 전환을 종료합니다.	EOT
이벤트 태스크의 실행을 트리거합니다.	EVENT
렁을 비활성화합니다.	AFI
로직에 자리 표시자를 삽입합니다.	NOP

추가 참조

계산/연산 명령어 페이지의 411

비교 명령어 페이지의 329

비트 명령어 페이지의 81

ASCII 문자열 명령어 페이지의 907

ASCII 변환 명령어 페이지의 927

항상 거짓(AFI)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

AFI 명령어는 EnableOut 을 거짓으로 설정합니다.

사용 가능한 언어

래더 다이어그램

펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

래더 다이어그램

없음

설명

AFI 명령어는 EnableOut 을 거짓으로 설정합니다.

연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

진한 실선 아래의 모든 조건은 정상 스캔 모드 중에만 수행될 수 있습니다.

조건	동작
사전 스캔	N/A
렁-입력-조건이 거짓임	EnableOut 을 거짓으로 해제합니다.
렁-입력-조건이 참임	EnableOut 을 거짓으로 해제합니다.
사후 스캔	N/A

예

래더 다이어그램

AFI 명령어를 사용하여 프로그램을 디버깅하는 동안 일시적으로 렁을 비활성화합니다. AFI 는 이 렁에 대한 모든 명령어를 비활성화합니다.

추가 참조

프로그램 제어 명령어 페이지의 680

마스터 제어 리셋(MCR) 페이지의 704

작업 없음(NOP) 페이지의 708

일시 종료(TND) 페이지의 715

공통 속성 페이지의 963

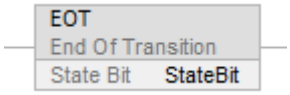
전환 끝(EOT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

EOT 명령어는 전환의 상태를 설정하기 위해 사용됩니다. 일반적으로 전환에서 호출된 서브루틴(JSR)에서 발생합니다. EOT 에 사용된 상태 비트 파라미터가 전환의 상태를 결정합니다. 상태 비트가 참으로 설정된 경우 SFC 가 다음 상태로 전환되고 그렇지 않은 경우 EOT 가 NOP 의 역할을 합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

EOT(StateBit);

피연산자

래더 다이어그램

피연산자	유형	형식	설명
State Bit	BOOL	태그	전환의 상태 (0 = 실행, 1 = 완료)

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
State Bit	BOOL	태그	전환의 상태 (0 = 실행, 1 = 완료)

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

설명

EOT 명령어는 부울 상태를 반환하므로 여러 SFC 루틴이 EOT 명령어를 포함한 같은 루틴을 공유할 수 있습니다. 호출 루틴이 전환이 아닌 경우 EOT 명령어는 NOP 명령어 역할을 합니다.

Logix 컨트롤러의 경우, 일부 Logix 프로그래밍 언어에서 렁 조건을 사용할 수 없으므로 반환 파라미터는 전환 상태를 반환합니다.

연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	이 명령어는 데이터 비트 값을 호출 루틴에 반환합니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	N/A
정상 실행	이 명령어는 데이터 비트 값을 호출 루틴에 반환합니다.
사후 스캔	N/A

예

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

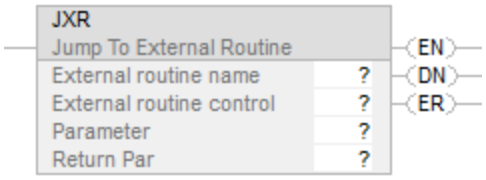
외부 루틴으로 이동(JXR)

이 정보는 SoftLogix 5800 컨트롤러에만 적용됩니다.

JXR 명령어는 외부 루틴을 실행합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에 사용할 수 없습니다.

피연산자

래더 다이어그램

피연산자	유형	형식	설명
External routine name	ROUTINE	Name	실행할 외부 루틴
External routine control	EXT_ROUTINE_CONTROL	태그	Control 구조
Parameter	BOOL SINT INT DINT REAL 구조	즉시 태그 배열 태그	외부 루틴의 변수에 복사할 이 루틴의 데이터 파라미터는 옵션입니다. 필요한 경우 여러 파라미터를 입력합니다. 파라미터를 10 개까지 지정할 수 있습니다.
Return parameter	BOOL SINT INT DINT REAL	태그	외부 루틴의 결과를 복사할 이 루틴의 태그 반환 파라미터는 옵션입니다. 반환 파라미터를 한 개만 지정할 수 있습니다.

EXT_ROUTINE_CONTROL 구조

니모닉	데이터 유형	설명	구현
ErrorCode	SINT	에러가 발생하면 이 값이 에러를 식별합니다. 유효값은 0 ~ 255 입니다.	미리 정의된 에러 코드는 없습니다. 외부 루틴의 개발자는 에러 코드를 제공해야 합니다.
NumParams	SINT	이 값은 이 명령어와 연결된 파라미터 수를 나타냅니다.	표시 전용임 - 이 정보는 명령어 입력에서 가져옵니다.

ParameterDefs	EXT_ROUTINE_PARAMETERS[10]	이 배열은 외부 루틴에 전달할 파라미터의 정의를 포함합니다. 이명령어는 파라미터를 10 개까지 전달할 수 있습니다.	표시 전용임 - 이 정보는 명령어 입력에서 가져옵니다.
ReturnParamDef	EXT_ROUTIN_PARAMETERS	이 값은 외부 루틴에서 오는 반환 파라미터의 정의를 포함합니다. 반환 파라미터는 한 개만 있습니다.	표시 전용임 - 이 정보는 명령어 입력에서 가져옵니다.
EN	BOOL	설정된 경우 활성화 비트는 JXR 명령어가 활성화되었음을 나타냅니다.	외부 루틴이 이 비트를 설정합니다.
ReturnsValue	BOOL	설정된 경우 이 비트는 반환 파라미터가 명령어에 대해 입력되었음을 나타냅니다. 해제된 경우 이 비트는 명령어에 대해 입력된 반환 파라미터가 없음을 나타냅니다.	표시 전용임 - 이 정보는 명령어 입력에서 가져옵니다.
DN	BOOL	완료 비트는 외부 루틴이 완료될 때까지 한 번 실행된 경우에 설정됩니다.	외부 루틴이 이 비트를 설정합니다.
ER	BOOL	에러 비트는 에러가 발생한 경우에 설정됩니다. 프로그램이 에러 비트를 지울 때까지 명령어가 실행을 중지합니다.	외부 루틴이 이 비트를 설정합니다.
FirstScan	BOOL	이 비트는 컨트롤러를 실행 모드로 전환한 후 첫 번째 스캔인지 여부를 식별합니다. 필요한 경우 FirstScan 을 사용하여 외부 루틴을 초기화합니다.	컨트롤러는 이 비트를 스캔 상태가 반영되도록 설정합니다.

<table>
<tr><td>EnableOut</td><td>BOOL</td><td>출력 활성화.</td><td>외부 루틴이 이 비트를 설정합니다.</td></tr>
<tr><td>EnableIn</td><td>BOOL</td><td>활성화 입력.</td><td>컨트롤러는 이 비트를 렁-입력-조건이 반영되도록 설정합니다. 명령어는 렁 조건과 상관없이 실행됩니다. 외부 루틴 개발자는 이 비트를 모니터링하고 그에 따라 조치해야 합니다.</td></tr>
<tr><td>User1
User0</td><td>BOOL
BOOL</td><td>이 비트는 사용자가 사용할 수 있습니다. 컨트롤러는 이 비트를 초기화하지 않습니다.</td><td>외부 루틴 또는 사용자 프로그램이 이 비트를 설정할 수 있습니다.</td></tr>
<tr><td>ScanType1
ScanType0</td><td>BOOL
BOOL</td><td>이 비트는 현재 스캔 유형을 식별합니다.
<table><tr><th>비트 값</th><th>스캔 유형</th></tr><tr><td>00</td><td>정상</td></tr><tr><td>01</td><td>사전 스캔</td></tr><tr><td>10</td><td>사후 스캔(릴레이 래더 다이어그램에는 적용되지 않음)</td></tr></table></td><td>컨트롤러는 이 비트를 스캔 상태가 반영되도록 설정합니다.</td></tr>
</table>

설명

외부 루틴으로 이동(JXR) 명령어를 사용하여 프로젝트의 래더 루틴에서 외부 루틴을 호출합니다. JXR 명령어는 래더 루틴과 외부 루틴 사이에 값을 전달할 수 있도록 여러 파라미터를 지원합니다.

JXR 명령어는 서브루틴으로 이동(JSR) 명령어와 유사합니다. JXR 명령어는 지정된 외부 루틴의 실행을 시작합니다.

- 외부 루틴은 한 번 실행합니다.
- 외부 루틴이 실행한 후 로직 실행은 JXR 명령어를 포함한 루틴으로 돌아갑니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트가 발생하는 원인	폴트 유형	폴트 코드:
외부 루틴 DLL 에서 예외가 발생합니다. DLL 을 로드할 수 없습니다. DLL 에서 진입점을 찾을 수 없습니다.	4	88

실행

JXR 은 DLL 구현에 따라 동기적일 수도 있고 비동기적일 수도 있습니다. 또한 DLL 의 코드는 스캔 상태에 응답하는 방법, 렁-입력-조건 상태 및 렁-출력-조건 상태를 결정합니다.

JXR 명령어 사용 및 외부 루틴 생성에 대한 자세한 내용은 SoftLogix5800 System User Manual(발행 번호 1789-UM002)을 참조하십시오.

추가 참조

공통 속성 페이지의 963

라벨로 이동(JMP) 및 라벨(LBL)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

JMP 및 LBL 명령어는 래더 로믹의 일부를 건너뜁니다.

사용 가능한 언어

래더 다이어그램

펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

래더 다이어그램

피연산자	유형	형식	설명
JMP 명령어			
Label name		라벨 이름	연결된 LBL 명령어에 대한 이름을 입력합니다.
LBL 명령어			
Label name		라벨 이름	실행이 참조 LBL 명령어로 이동합니다.

설명

참인 경우 JMP 명령어는 참조한 LBL 명령어로 이동하며 컨트롤러는 거기서부터 실행을 계속합니다. 거짓인 경우 JMP 명령어는 래더 실행에 영향을 미치지 않습니다.

이 명령어가 참조하는 JMP 및 LBL 은 같은 루틴에 있어야 합니다.

JMP 명령어는 래더 실행을 앞으로 또는 뒤로 이동할 수 있습니다. 라벨까지 앞으로 이동하면 로직 세그먼트를 필요할 때까지 생략하여 프로그램 스캔 시간을 절약합니다. 뒤로 이동하면 컨트롤러가 로직의 반복을 되풀이할 수 있습니다.

중요: 너무 많은 횟수에 걸쳐 뒤로 이동하지 않도록 주의하십시오. 스캔이 제시간에 완료되지 않아 워치독 타이머가 타임아웃할 수 있습니다.

이동한 로직은 스캔되지 않습니다. 중요 로직을 이동한 영역 밖에 놓으십시오.

JMP 명령어는 연결된 라벨이 다음을 수행하기 전에 존재할 것을 요구합니다.

- 오프라인 작업할 때 다운로드
- 온라인 작업할 때 편집 내용 수락

LBL 명령어는 렁에서 첫 번째 명령어여야 합니다.

라벨 이름은 루틴 내에서 고유해야 합니다. 이름은 다음과 같을 수 있습니다.

- 최대 40 문자까지 포함
- 영문자, 숫자 및 밑줄(_) 포함

연산 상태 플래그에 영향

아니요.

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

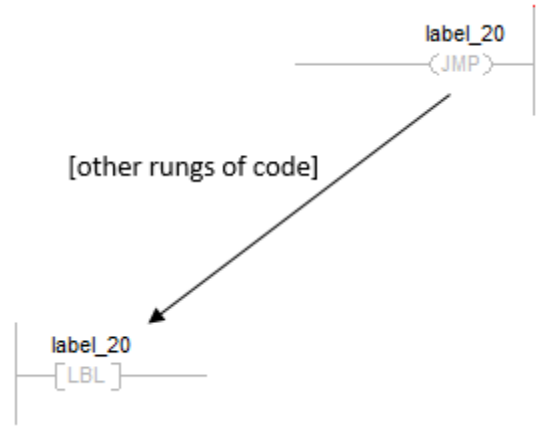
조건	동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	(JMP 의 경우)실행이 참조한 라벨 이름을 가진 LBL 명령어를 포함한 렁으로 이동합니다. (LBL 의 경우)아무 조치도 취하지 않습니다.
사후 스캔	N/A

예

래더 다이어그램

JMP

JMP 명령어가 활성화된 경우 실행은 label_20 과 함께 LBL 명령어를 포함한 렁에 도달할 때 까지 연속적인 로직의 렁을 건너뜁니다.



LBL

추가 참조

프로그램 제어 명령어 페이지의 680

서브루틴으로 이동(JSR), 서브루틴(SBR) 및 반환(RET) 페이지의 694

반복(FOR) 페이지의 730

중단(BRK) 페이지의 727

공통 속성 페이지의 963

서브루틴으로 이동(JSR), 서브루틴(SBR) 및 반환(RET)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

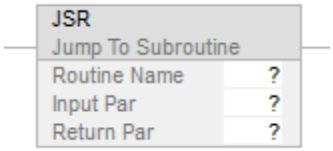
JSR 명령어는 다른 루틴을 호출합니다. 해당 루틴이 완료되면 실행이 JSR 명령어로 복귀합니다.

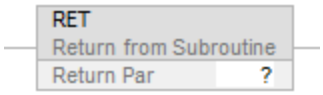
SBR 명령어는 JSR 에 의해 전달된 입력 파라미터를 수신합니다.

RET 명령어는 반환 파라미터를 JSR 에 다시 전달하고 서브루틴의 스캔을 종료합니다.

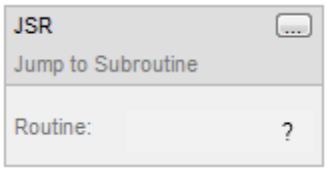
사용 가능한 언어

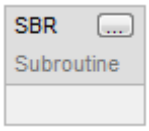
래더 다이어그램

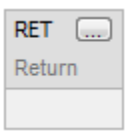




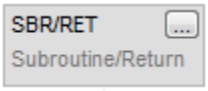
평션 블록







순차 평션 차트



ST(스트럭처드 텍스트)

JSR(RoutineName,InputCount,InputPar,ReturnPar);

SBR(InputPar);

RET(ReturnPar);

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

SBR 또는 RET 명령어의 각 파라미터에 JSR 명령어의 해당 파라미터와 같은 데이터 유형(모든 배열 차원 포함)을 사용합니다. 다른 데이터 유형을 사용하면 예기치 않은 결과가 발생할 수 있습니다.

래더 다이어그램

JSR 명령어

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
Routine Name	ROUTINE	ROUTINE	이름	실행할 서브루틴

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
Input Par	BOOL SINT INT DINT REAL 구조	BOOL SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL 구조	즉시 태그 배열 태그	이 루틴에서 서브루틴의 태그에 복사할 데이터. • 입력 파라미터는 옵션임 • 필요한 경우 입력 파라미터를 최대 40 개까지 입력합니다.
Return Par	BOOL SINT INT DINT REAL 구조	BOOL SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL 구조	태그 배열 태그	이 루틴에 서브루틴의 결과를 복사하도록 태그를 설정합니다. • 반환 파라미터는 옵션임 • 필요한 경우 반환 파라미터를 최대 40 개까지 입력

SBR 명령어

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
Input Par	BOOL SINT INT DINT REAL 구조	BOOL SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL 구조	태그 배열 태그	• 이 루틴의 태그이며, 이에 JSR 명령어의 해당 입력 파라미터(최대 40 개)를 복사.

RET 명령어

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
Return Par	BOOL SINT INT DINT REAL 구조	BOOL SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL 구조	즉시 태그 배열 태그	JSR 명령어의 해당 반환 파라미터(최대 40 개)에 복사할 이 루틴의 데이터.

연산 상태 플래그에 영향

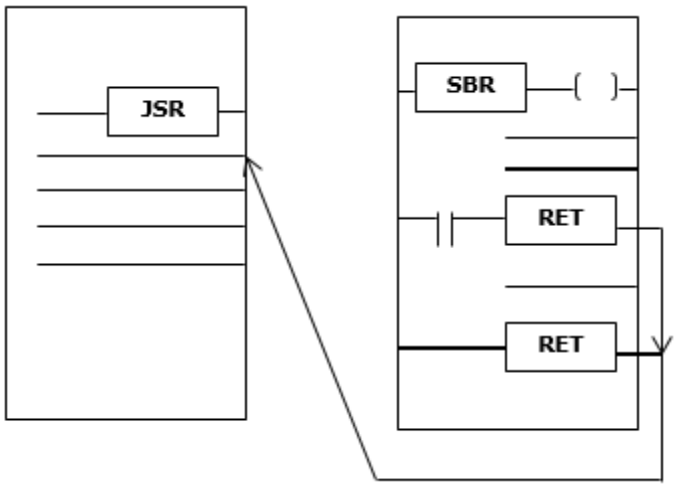
아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
JSR 명령어는 SBR 명령어보다 입력 파라미터 수가 적음	4	31
JSR 명령어가 폴트 루틴으로 이동	4	990 또는 사용자 제공
RET 명령어는 JSR 명령어보다 입력 파라미터 수가 적음	4	31
주 루틴에는 RET 명령어가 포함	4	31

연산

중요:	어떤 루틴이든 JSR 명령어를 포함할 수 있지만 JSR 명령어는 주 루틴을 호출(실행)할 수 없습니다.



JSR 명령어는 지정된 루틴(서브루틴이라 함)의 실행을 시작합니다.

- 서브루틴은 스캔될 때마다 실행합니다.
- 서브루틴이 실행한 후 로직 실행은 JSR 명령어를 포함한 루틴으로 돌아가서 JSR 다음의 명령어를 사용하여 계속합니다.

서브루틴에 대한 이동을 프로그래밍하려면 다음 지침을 따릅니다.

JSR

- 서브루틴의 태그에 데이터를 복사하려면 입력 파라미터를 입력합니다.

- 서브루틴의 결과를 이 루틴의 태그에 복사하려면 반환 파라미터를 입력합니다.
- 필요에 따라 입력을 40 개까지 입력하며 반환 파라미터를 40 개까지 입력합니다.

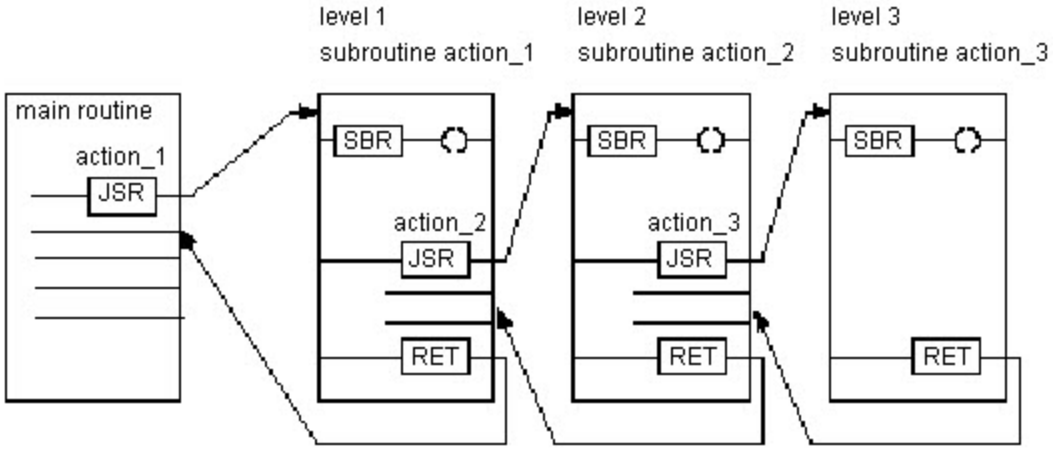
SBR

- JSR 명령어가 입력 파라미터를 포함한 경우 SBR 명령어를 입력합니다.
- SBR 명령어를 루틴의 첫 번째 명령어로 놓습니다.
- JSR 명령어의 각 입력 파라미터에 대해 데이터를 복사할 태그를 입력합니다.

RET

- JSR 명령어가 반환 파라미터를 포함한 경우 RET 명령어를 입력합니다.
- RET 명령어를 루틴의 마지막 명령어로 놓습니다.
- JSR 명령어의 각 반환 파라미터에 대해 JSR 명령어에 전달할 반환 파라미터를 입력합니다.
- 래더 루틴에서는 필요한 경우 다른 입력 조건을 기반으로 서브루틴을 종료하기 위해 추가 RET 명령어를 놓습니다(펑션 블록 루틴은 RET 명령어를 한 개만 허용).

최대 40 개의 파라미터를 서브루틴에 전달하고 최대 40 개의 파라미터가 서브루틴에서 반환되는 상태에서 중첩된 서브루틴을 25 개까지 호출합니다.



팁: 가변 피연산자를 추가 및 제거하려면 **편집 > 래더 요소 편집(Edit > Edit Ladder Element)** 메뉴를 선택합니다. JSR 및 SBR 명령어의 경우 입력 파라미터를 추가합니다. JSR 및 RET 명령어의 경우 출력 파라미터를 추가합니다. 3 개 명령어에 모두 명령어 파라미터를 제거합니다.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	렁은 거짓으로 설정됩니다. 컨트롤러가 모든 서브루틴을 실행합니다. 서브루틴의 모든 렁이 사전 스캔되도록 하기 위해 컨트롤러가 RET 명령어를 무시합니다(즉, RET 명령어가 서브루틴을 종료하지 않음). 입력 및 반환 파라미터는 전달되지 않습니다. 같은 서브루틴이 여러 번 호출되면 한 번만 사전 스캔됩니다.
렁-입력-조건이 (JSR 명령어에 대해) 거짓임	N/A
렁-입력-조건이 참임	파라미터가 전달되고 서브루틴이 실행됩니다.
사후 스캔	사전 스캔과 같은 조치

평션 블록

조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
EnableIn 이 거짓임	N/A
EnableIn 이 참임	파라미터가 전달되고 서브루틴이 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 참조하십시오.

ST(스트럭처드 텍스트)

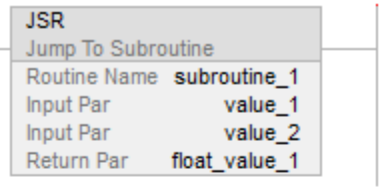
조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	파라미터가 전달되고 서브루틴이 실행됩니다.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 참조하십시오.

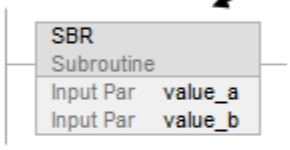
예제

예 1

래더 다이어그램

When enabled, the JSR instruction passes value_1 and value_2 to routine_1.

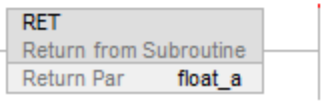




The SBR instruction receives value_1 and value_2 from the JSR instruction and copies those values to value_a and value_b, respectively. Logic execution continues in this routine.

[other rungs of code]

When enabled, the RET instruction sends float_a to the JSR instruction. The JSR instruction receives float_a and copies the value to float_value_1. Logic execution continues with the next instruction following the JSR instruction.



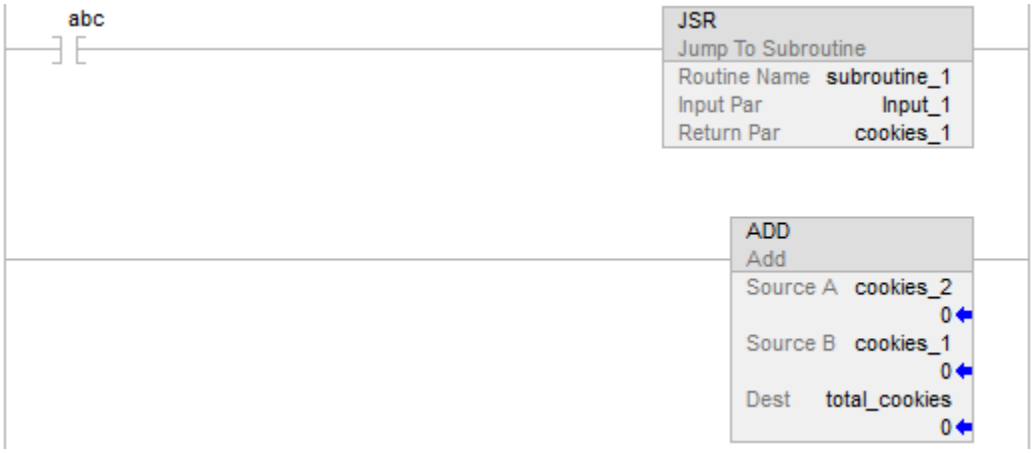
ST(스트럭처드 텍스트)

루틴	Program
주 루틴	JSR(routine_1,2,value_1,value_2,float_value_1);
서브루틴	SBR(value_a,value_b); <statements>; RET(float_a);

예 2

래더 다이어그램

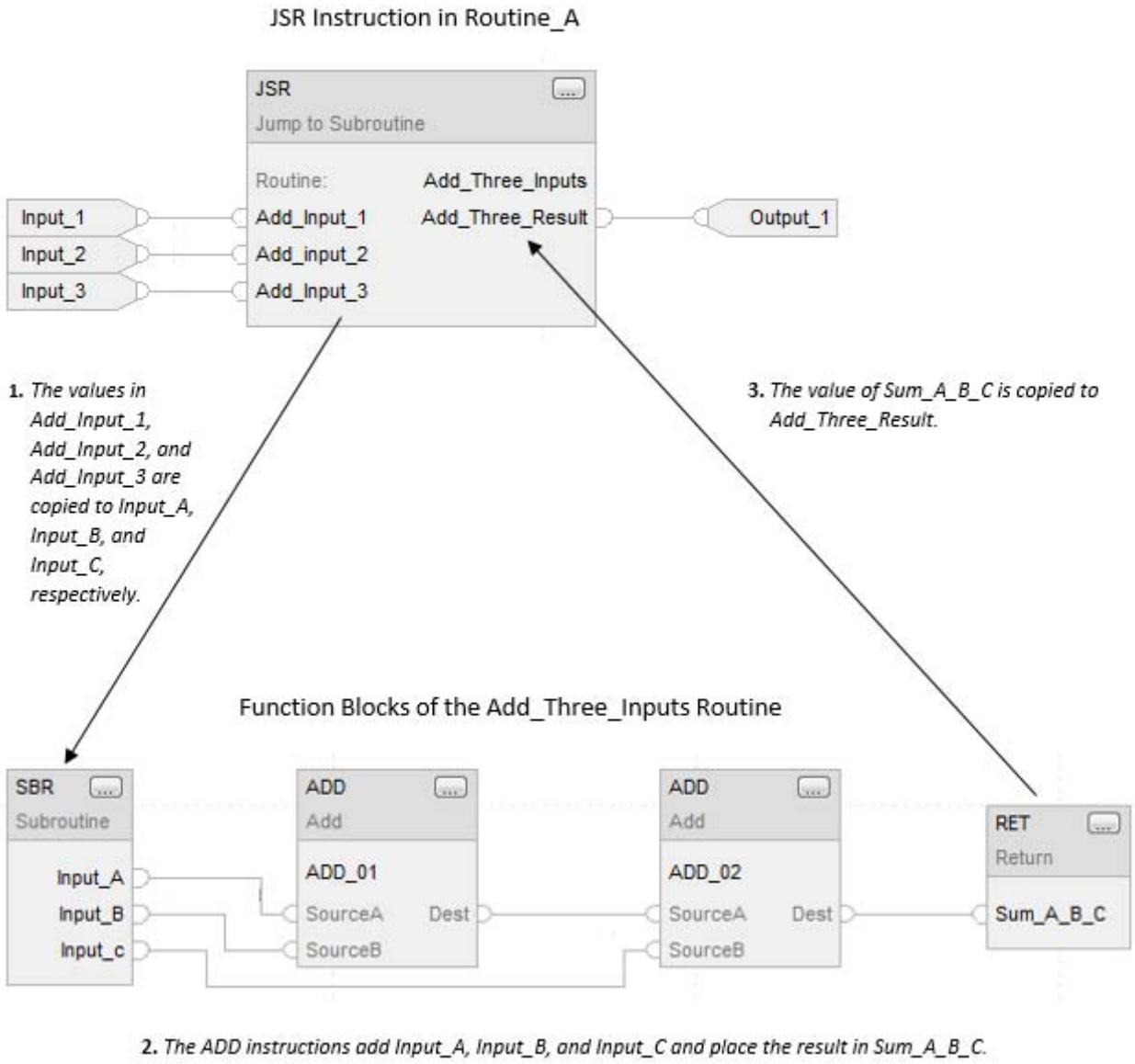
주 루틴



subroutine_1

예 3

평션 블록



추가 참조

[프로그램 제어 명령어](#) 페이지의 680

[배열을 통한 인덱스](#) 페이지의 978

[즉시 값](#) 페이지의 967

마스터 제어 리셋(MCR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

MCR 명령어는 마스터 제어 릴레이(시리얼 연결한 비상 정지 스위치에 의해 비활성화할 수 있는 고정 배선된 의무 릴레이)를 시뮬레이션합니다. 릴레이 전원이 차단될 때마다 모든 응용 프로그램 I/O 장치의 전원을 차단하기 위해 접점이 개방됩니다. MCR 명령어는 렁의 섹션을 선택적으로 비활성화할 수 있습니다.

사용 가능한 언어

래더 다이어그램

펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

설명

MCR 명령어는 렁의 정상 동작을 오버라이드할 수 있으며, 모든 명령어를 렁-입력-조건이 거짓인 것처럼 강제 실행하게 할 수 있습니다. 일반적으로 명령어의 거짓 실행은 참 실행보다 빠르므로 코드의 불필요한 섹션을 선택적으로 비활성화하면 스캔 시간이 전체적으로 감소됩니다.

MCR 명령어가 렁-입력-조건 거짓을 사용하여 실행할 때마다 오버라이드 동작이 전환됩니다. 따라서 보통은 MCR 명령어 두 개가 필요합니다. 하나는 "구역"을 실행하고 두 번째 명령어는 이를 종료합니다.

시작 MCR 은 일반적으로 하나 이상의 입력 명령어에 의해 조건이 지정됩니다. 입력 조건이 거짓이면 구역이 비활성화됩니다. 입력 조건이 참이면 구역은 정상 작동합니다.

종료 MCR 은 일반적으로 무조건입니다. 구역이 활성화되면 종료 MCR 이 참이므로 아무 작업도 수행하지 않습니다. 그러나 구역이 비활성화되면 종료 MCR 이 거짓이므로 오버라이드를 전환하여 그 다음의 렁을 다시 활성화합니다.

MCR 구역을 프로그래밍하는 경우 다음 사항에 유의하십시오.

MCR 명령어는 렁에서 마지막 명령어여야 합니다.

- 무조건 MCR 명령어를 사용하여 구역을 종료해야 합니다. 종료 MCR 이 거짓이고 구역이 활성화된 경우 종류 MCR 은 그 다음의 모든 렁을 비활성화합니다.
- 한 MCR 구역을 다른 MCR 구역에 중첩할 수 없습니다. 각 프로그램에 오버라이드 비트 한 개만 있습니다. 각 MCR 명령어는 이 오버라이드를 전환할 수 있습니다. MCR 구역의 중첩을 시도하면 실제로 여러 개의 더 작은 영역이 생성됩니다.
- MCR 구역으로 이동하지 마십시오. 시작 MCR 이 실행되지 않으면 구역이 비활성화되지 않습니다.
- 오버라이드 비트는 루틴의 끝에서 자동으로 리셋합니다. MCR 구역이 루틴의 끝까지 계속하는 경우 구역을 종료하는 MCR 명령어를 프로그래밍하지 않아도 되지만, 온라인 편집 시 혼동을 피하기 위해 종료 MCR 을 항상 사용하는 것이 좋습니다.

MCR 이 서브루틴 또는 AOI 에서 비활성화되면 오버라이드 비트는 서브루틴/AOI 가 반환할 때 초기화됩니다.

AOI 는 AOI 가 호출될 때 초기화되는 자체 오버라이드 비트를 가지고 있습니다. AOI 가 비활성화된 MCR 구역에서 호출되면 거짓 스캔 모드 루틴이 정상적으로 실행합니다. AOI 가 반환한 후 구역의 상태는 AOI 가 호출되기 전의 상태로 복원됩니다.

중요: MCR 명령어는 비상 정지 기능을 제공하는 고정 배선된 마스터 제어 릴레이를 대체하지 않습니다. 비상 I/O 전원 차단을 제공하려면 여전히 고정 배선된 마스터 제어 릴레이를 설치해야 합니다.

중요: MCR 영역을 중복하거나 중첩하지 마십시오. 각 MCR 영역은 분리되고 완전해야 합니다. 이들이 중복 또는 중첩된 경우 예측할 수 없는 기계 작동이 발생하며 장비 손상 또는 인원 부상이 예상됩니다.

중요 연산은 MCR 영역 밖에 놓으십시오. MCR 구역에서 타이머 같은 명령어를 시작하면 명령어 실행은 구역이 비활성화되고 타이머가 해제될 때 거짓이 됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	오버라이드 동작이 전환되어 그 다음 렁을 활성화 또는 비활성화합니다.
렁-입력-조건이 참임	N/A
사후 스캔	N/A

예

래더 다이어그램

첫 번째 MCR 명령어가 활성화되면(input_1, input_2 및 input_3 이 설정됨), 컨트롤러는 MCR 구역(두 MCR 명령어 사이)의 렁을 실행하고 입력 조건에 따라 출력을 설정하거나 해제합니다.

첫 번째 MCR 명령어가 비활성화되면(모든 input_1, input_2 및 input_3 이 설정되지 않음), 컨트롤러는 MCR 구역(두 MCR 명령어 사이)의 렁을 실행하고 입력 조건과 상관없이 EnableIn 은 MCR 구역의 모든 렁에 대해 거짓이 됩니다.

추가 참조

프로그램 제어 명령어 페이지의 680

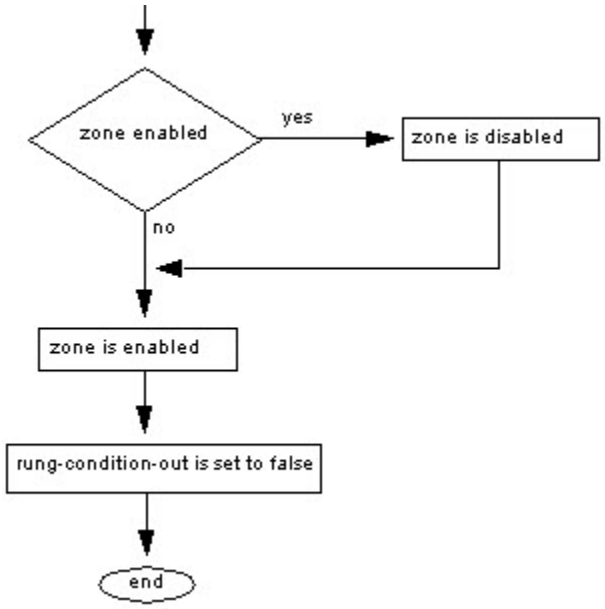
항상 거짓(AFI) 페이지의 682

작업 없음(NOP) 페이지의 708

일시 종료(TND) 페이지의 715

공통 속성 페이지의 963

MCR 흐름도(거짓)



작업 없음(NOP)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

NOP 명령어는 자리 표시자 기능을 합니다.

사용 가능한 언어

래더 다이어그램

펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

래더 다이어그램

없음

설명

NOP 명령어를 렁의 어디에나 놓을 수 있습니다. 활성화되면 NOP 명령어가 아무 작업도 수행하지 않습니다. 비활성화되면 NOP 명령어가 아무 작업도 수행하지 않습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	N/A
사후 스캔	N/A

예

래더 다이어그램

추가 참조

프로그램 제어 명령어 페이지의 680

항상 거짓(AFI) 페이지의 682

마스터 제어 리셋(MCR) 페이지의 704

일시 종료(TND) 페이지의 715

공통 속성 페이지의 963

SFC 일시 중지(SFP)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

SFP 명령어는 SFC 루틴을 일시 중지합니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

SFP(SFCRoutineName,TargetState);

피연산자

래더 다이어그램

피연산자	유형	형식	설명
SFCRoutineName	ROUTINE	이름	일시 중지할 SFC 루틴
TargetState	DINT	즉시	한 개 선택: • 실행 중(또는 0 입력) • 일시 중지됨(또는 1 입력)

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
SFCRoutineName	ROUTINE	이름	일시 중지할 SFC 루틴
TargetState	DINT	즉시	한 개 선택: • 실행 중(또는 0 입력) • 일시 중지됨(또는 1 입력)

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

설명

SFP 명령어를 사용하여 실행 중인 SFC 루틴을 일시 중지할 수 있습니다.

연산 상태 플래그에 영향

아니요

폴트 조건

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
루틴 유형이 SFC 루틴이 아님	4	85

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

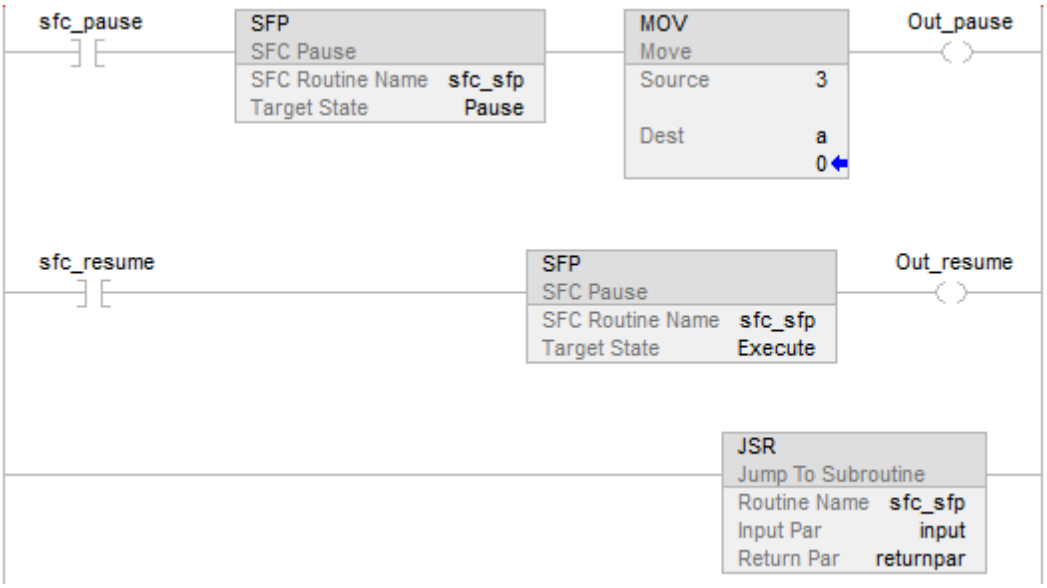
조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임.	N/A
렁-입력-조건이 참임	이 명령어는 지정된 SFC 루틴의 실행을 일시 중지하거나 다시 시작합니다.
사후 스캔	N/A

ST(스트럭쳐드 텍스트)

조건/상태	취해진 조치
사전 스캔	N/A
정상 실행	이 명령어는 지정된 SFC 루틴의 실행을 일시 중지하거나 다시 시작합니다.
사후 스캔	N/A

예

래더 다이어그램



추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

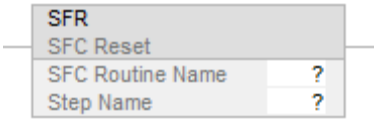
SFC 리셋(SFR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

SFR 명령어는 지정된 단계에서 SFC 루틴의 실행을 리셋합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

SFR(SFCRoutineName,StepName);

피연산자

래더 다이어그램

피연산자	유형	형식	설명
SFCRoutineName	ROUTINE	이름	리셋할 SFC 루틴
StepName	SFC_STEP	태그	실행을 다시 시작할 목표 단계

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
SFCRoutineName	ROUTINE	이름	리셋할 SFC 루틴
StepName	SFC_STEP	태그	실행을 다시 시작할 목표 단계

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

설명

SFR 명령어가 활성화된 경우:

- 지정된 SFC 루틴에서 모든 저장된 동작이 실행을 중지합니다(리셋).
- SFC 가 지정된 단계에서 실행을 시작합니다.
- 목표 단계가 0 인 경우 차트가 초기 단계로 재설정됩니다.

SFR 명령어의 Logix 구현은 PLC-5 컨트롤러의 구현과 다릅니다. PLC-5 컨트롤러에서 SFR 은 렁 조건이 참일 때 실행합니다. 리셋 후 SFC 는 SFR 을 포함한 렁이 거짓이 될 때까지 일시 중지된 상태로 유지합니다. 이를 통해 리셋 후 실행을 지연할 수 있습니다. PLC-5 SFR 명령어의 이 일시 중지/일시 중지 해제 기능은 렁 조건에서 해체되어 SFP 명령어로 이동됩니다.

연산 상태 플래그에 영향

아니요

폴트 조건

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
루틴 유형이 SFC 루틴이 아님	4	85
지정된 목표 단계가 SFC 루틴에 존재하지 않음	4	89

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	지정된 SFC 루틴 실행을 특정 단계로 리셋합니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	N/A
정상 실행	지정된 SFC 루틴 실행을 특정 단계로 리셋합니다.
사후 스캔	N/A

예

래더 다이어그램

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

일시 종료(TND)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

TND 명령어는 조건부로 루틴을 종료합니다.

사용 가능한 언어

래더 다이어그램

펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

TND();

피연산자

래더 다이어그램

없음

ST(스트럭처드 텍스트)

없음

설명

활성화되면 TND 명령어는 루틴의 종료 역할을 합니다. TND 명령어가 서브루틴 안에 있으면 제어권은 호출 루틴에 돌아갑니다. TND 명령어가 주 루틴 안에 있으면 제어는 현재 태스크 내의 다음 프로그램으로 돌아갑니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임.	루틴 종료
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표의 사후 스캔을 참조하십시오.

ST(스트럭처드 텍스트)

```
InputA[:=] OutputB;

IF (InputA) THEN

  TND();

END_IF;


InputE [:=] OutputF;
```

추가 참조

프로그램 제어 명령어 페이지의 680

항상 거짓(AFI) 페이지의 682

마스터 제어 리셋(MCR) 페이지의 704

작업 없음(NOP) 페이지의 708

공통 속성 페이지의 963

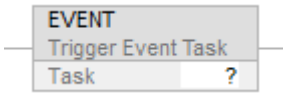
트리거 이벤트 태스크(EVENT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

EVENT 명령어는 이벤트 태스크의 실행 한 개를 트리거합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

EVENT(task_name);

피연산자

래더 다이어그램

피연산자	유형	형식	설명
Task	TASK	이름	실행할 이벤트 태스크. 이벤트 태스크가 아닌 태스크를 지정하면 지정한 태스크는 실행되지 않습니다.

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Task	TASK	이름	실행할 이벤트 태스크. 이벤트 태스크가 아닌 태스크를 지정하면 지정한 태스크는 실행되지 않습니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

설명

EVENT 명령어를 사용하여 이벤트 태스크를 프로그래밍 방식으로 실행합니다.

명령어가 실행할 때마다 지정된 이벤트 태스크를 트리거합니다.

이벤트 태스크를 다시 트리거하기 전에 실행을 완료하기에 충분한 시간을 부여해야 합니다. 그렇지 않으면 중복이 발생합니다.

이벤트 태스크가 이미 실행하고 있는 동안 EVENT 명령어를 실행하면 컨트롤러는 중복 카운터를 증가시키지만 이벤트 태스크를 트리거하지 않습니다.

EVENT 명령어를 사용하면 모든 트리거 유형으로 이벤트 태스크를 트리거할 수 있습니다.

프로그래밍 방식으로 EVENT 명령어가 태스크를 트리거했는지 여부를 결정합니다.

EVENT 명령어가 이벤트 태스크를 트리거했는지 여부를 결정하려면 시스템 값 가져오기(GSV) 명령어를 사용하여 태스크의 Status 속성을 모니터링합니다.

<table>
<tr><th>속성</th><th>데이터 유형</th><th>명령어</th><th colspan="2">설명</th></tr>
<tr><td rowspan="5">Status</td><td rowspan="5">DINT</td><td rowspan="5">GSV
SSV</td><td colspan="2">태스크에 대한 상태 정보 제공함. 컨트롤러가 비트를 설정한 후 수동으로 비트를 해제하여 해당 유형의 다른 폴트가 발생했는지 여부를 결정해야 합니다.</td></tr>
<tr><td>다음의 여부를 결정하려면</td><td>이 비트를 검사</td></tr>
<tr><td>EVENT 명령어가 태스크를 트리거함(이벤트 태스크만 해당)</td><td>0</td></tr>
<tr><td>타임아웃에 의해 태스크가 트리거됨(이벤트 태스크만 해당)</td><td>1</td></tr>
<tr><td>이 태스크에 대한 중복이 발생함</td><td>2</td></tr>
</table>

컨트롤러는 Status 속성의 비트가 설정된 후 해당 비트를 해제하지 않습니다. 새 상태 정보를 위해 비트를 사용하려면 해당 비트를 수동으로 지워야 합니다. 시스템 값 설정(SSV) 명령어를 사용하여 속성을 다른 값으로 설정합니다.

연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건	취해진 조치
사전 스캔	N/A
정상 실행	명령어가 실행됩니다.
사후 스캔	N/A

예

예 1

컨트롤러는 여러 프로그램을 사용하지만 공통의 종료 절차를 사용합니다. 각 프로그램은 종료해야 하는 조건을 감지한 경우에 켜지는 Shut_Down_Line 이라는 프로그램 범위 태그를 사용합니다. 각 프로그램의 로직은 다음과 같이 실행됩니다.

Shut_Down_Line = on(조건이 종료를 요구함)이면

Shut_Down 태스크를 한 번 실행

래더 다이어그램

프로그램 A

프로그램 B

ST(스트럭처드 텍스트)

프로그램 A

IF Shut_Down_Line AND NOT Shut_Down_Line_One_Shot THEN

EVENT (Shut_Down);

END_IF;

Shut_Down_Line_One_Shot:=Shut_Down_Line;

프로그램 B

IF Shut_Down_Line AND NOT Shut_Down_Line_One_Shot THEN

EVENT (Shut_Down);

END_IF;

Shut_Down_Line_One_Shot:=Shut_Down_Line;

예 2

다음 예는 EVENT 명령어를 사용하여 이벤트 태스크를 초기화합니다. 다른 유형의 이벤트는 일반적으로 이벤트 태스크를 트리거합니다.

연속 태스크

IF Initialize_Task_1 = 1 THEN

ONS 명령어는 EVENT 명령어의 실행을 1 스캔으로 제한합니다.

EVENT 명령어는 Task_1(이벤트 태스크)의 실행을 트리거합니다.

Task_1(이벤트 태스크)

GSV 명령어는 Task_Status(DINT 태그) = 이벤트 태스크의 상 Status 속성을 설정합니다. Instance Name 속성에서 THIS 는 명령어가 있는 태스크(예, Task_1)에 대한 TASK 객체를 의미합니다.

Task_Status.0 = 1 이면 EVENT 명령어가 이벤트 태스크를 트리거했습니다(즉, 연속 태스크가 해당 EVENT 명령어를 실행하여 이벤트 태스크를 초기화함).

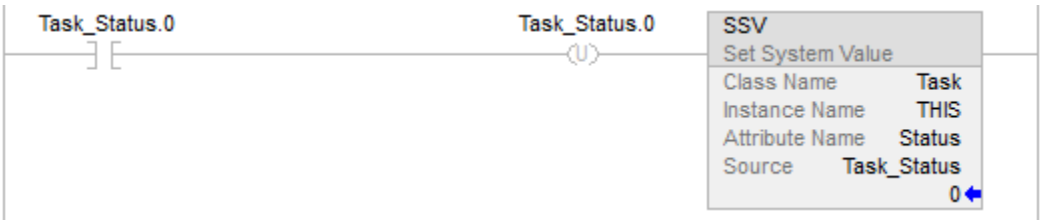
RES 명령어는 이벤트 태스크가 사용하는 카운터를 리셋합니다.

컨트롤러는 Status 속성의 비트가 설정된 후 해당 비트를 해제하지 않습니다. 새 상태 정보를 위해 비트를 사용하려면 해당 비트를 수동으로 지워야 합니다.

Task_Status.0 = 1 이면 해당 비트를 해제합니다.

OTU 명령어는 Task_Status.0 = 0 을 설정합니다.

SSV 명령어는 THIS 태스크(Task_1)의 Status 속성 = Task_Status 를 설정합니다. 이는 해제된 비트를 포함합니다.



추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

사용자 중단 비활성화(UID)/사용자 중단 활성화(UIE)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

UID 명령어 및 UIE 명령어는 소수의 중요 렁이 다른 태스크에 의해 중단되는 것을 방지하기 위해 함께 작동합니다.

사용 가능한 언어

래더 다이어그램

평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

UID();

UIE();

피연산자

래더 다이어그램

이 명령어는 래더 다이어그램에 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다. 피연산자가 없더라도 괄호 ()를 명령어 니모닉 뒤에 입력해야 합니다.

설명

렁-입력-조건이 참인 경우 다음과 같이 됩니다.

- UID 명령어는 우선 순위가 더 높은 태스크이 현재 태스크를 중단하는 것을 방지하지만 폴트 루틴 또는 컨트롤러 폴트 처리기의 실행을 비활성화하지 않습니다.

- UIE 명령어는 현재 태스크를 중단하기 위해 다른 태스크을 활성화합니다.

일련의 렁이 중단되는 것을 방지하려면:

1. 차단되기를 원하지 않는 렁 수를 가능하면 적게 제한합니다. 인터럽트를 장시간 동안 비활성화하면 통신 손실이 발생할 수 있습니다.
2. 중단되기를 원하지 않는 첫 번째 렁 위에 렁 및 UID 명령어를 입력합니다.
3. 중단되기를 원하지 않는 시리즈의 마지막 렁 뒤에 렁 및 UIE 명령어를 입력합니다.
4. 필요한 경우 UID/UIE 명령어 쌍을 중첩할 수 있습니다.

UID 가 처음 호출될 때에는 우선 순위를 급격히 전환하고 이전 우선 순위를 저장하고 중첩 카운터를 증가시킵니다. 각 이후 호출은 카운트를 증가시킵니다. UIE 는 중첩 카운터를 감소시킵니다. 새 값이 0 인 경우 저장된 우선 순위를 복원합니다.

연산 상태 플래그에 영향

아니요.

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

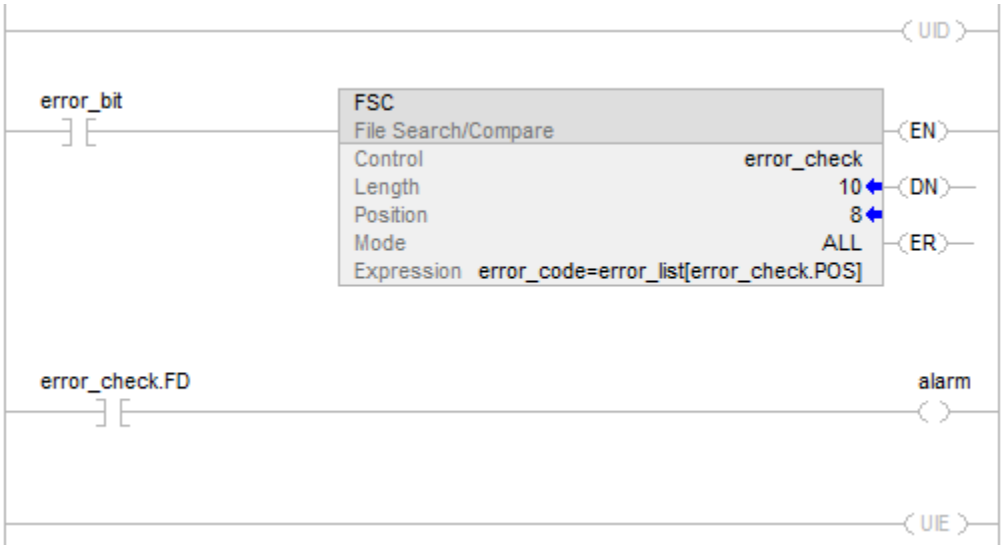
조건/상태	동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	UID 명령어는 포함 사용자 태스크가 중단되는 것을 방지합니다. UIE 명령어는 중단할 포함 사용자 태스크를 보통의 경우와 같이 활성화합니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건/상태	동작
사전 스캔	N/A
정상 실행	UID 명령어는 포함 사용자 태스크가 중단되는 것을 방지합니다. UIE 명령어는 중단할 포함 사용자 태스크를 보통의 경우와 같이 활성화합니다.
사후 스캔	N/A

예

래더 다이어그램



ST(스트럭처드 텍스트)

```
UID();

<statements>

UIE();
```

추가 참조

프로그램 제어 명령어 페이지의 680

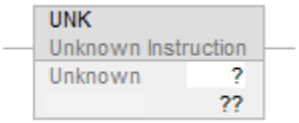
공통 특성 페이지의 963

알 수 없는 명령어(UNK)

UNK 명령어는 Logix Designer 명령어 집합 내에 정의되지 않은 명령어 유형을 입력했다는 표시의 기능을 합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 펑션 블록에서 사용할 수 없습니다.

피연산자

래더 다이어그램

피연산자	유형	형식	설명
알 수 없음	즉시	즉시	

추가 참조

프로그램 제어 명령어 페이지의 680

반복/중단 명령어

반복/중단 명령어

반복적으로 서브루틴을 호출하려면 FOR 명령어를 사용합니다. 서브루틴의 실행을 중단하려면 BRK 명령어를 사용합니다.

사용 가능한 명령어

래더 다이어그램

FOR	BRK

반복적으로 서브루틴을 호출하려면 FOR 명령어를 사용합니다. 서브루틴의 실행을 중단하려면 BRK 명령어를 사용하십시오.

실행할 작업:	사용할 명령어
루틴을 반복해서 실행합니다.	반복(FOR)
루틴의 반복 실행을 종료합니다.	중단(BRK)
FOR 명령어로 돌아갑니다.	반환(RET)

추가 참조

계산/연산 명령어 페이지의 411

비교 명령어 페이지의 329

비트 명령어 페이지의 81

ASCII 문자열 명령어 페이지의 907

ASCII 변환 명령어 페이지의 927

중단(BRK)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380,

CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

BRK 명령어는 FOR 명령어에 의해 호출된 루틴의 실행을 중단합니다.

사용 가능한 언어

래더 다이어그램

펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

설명

활성화된 경우 BRK 명령어는 루틴을 종료하고 제어권을 최근 실행한 FOR 명령어가 포함된 루틴에 반환하고 해당 명령어 다음에 실행을 다시 시작합니다. 이 스캔 중에 해당 실행에서 이 BRK 명령어 앞에 FOR 명령어가 없으면 BRK 는 아무 작업도 실행하지 않습니다.

중첩 FOR 명령어가 있는 경우 BRK 명령어는 제어권을 맨 안쪽의 FOR 명령어에 반환합니다.

연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

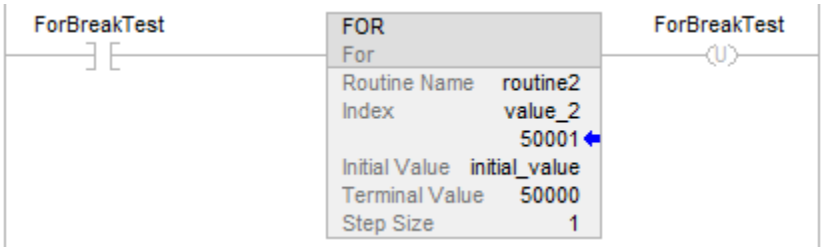
래더 다이어그램

조건/상태	동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

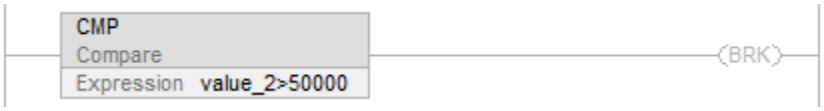
예

활성화된 경우 BRK 명령어는 현재 루틴의 실행을 중지하고 호출 FOR 명령어 다음의 명령어로 돌아갑니다.

래더 다이어그램



이는 routine2 임:



추가 참조

공통 특성 페이지의 963

반복/중단 명령어 페이지의 727

반복(FOR) 페이지의 730

라벨로 이동(JMP) 및 라벨(LBL) 페이지의 690

서브루틴으로 이동(JSR), 서브루틴(SBR) 및 반환(RET)
페이지의 694

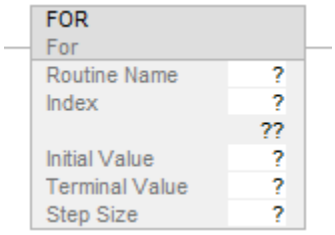
반복(FOR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

FOR 명령어는 루틴을 반복적으로 실행합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

래더 다이어그램

피연산자	유형	형식	설명
Routine name	ROUTINE	태그	FOR 루프가 실행될 때마다 호출되는 서브루틴입니다.
Index	DINT	태그	루틴이 실행된 횟수를 계산합니다.
Initial value	SINT INT DINT	즉시 태그	인덱스를 시작할 값
Terminal value	SINT INT DINT	즉시 태그	루틴 실행을 중지할 값

Step size	SINT INT DINT	즉시 태그	FOR 명령어가 루틴을 실행할 때마다 인덱스에 추가할 양

설명

활성화된 경우 FOR 명령어는 Index 가 Terminal value 를 초과할 때까지 루틴을 반복적으로 실행합니다. 이 명령어는 파라미터를 루틴에 전달하지 않습니다.

단계 값은 양수 또는 음수일 수 있습니다. 음수일 경우 인덱스가 터미널 값보다 작으면 루프가 종료됩니다. 양수일 경우 인덱스가 터미널 값보다 클 때 루프가 종료됩니다.

FOR 명령어가 루틴을 실행할 때마다 인덱스에 단계 크기가 추가됩니다.

단일 스캔에서 너무 많이 루프하지 않도록 주의하십시오. 과도한 반복 횟수로 인해 컨트롤러의 워치독이 타임아웃되어 메이저 폴트가 발생할 수 있습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

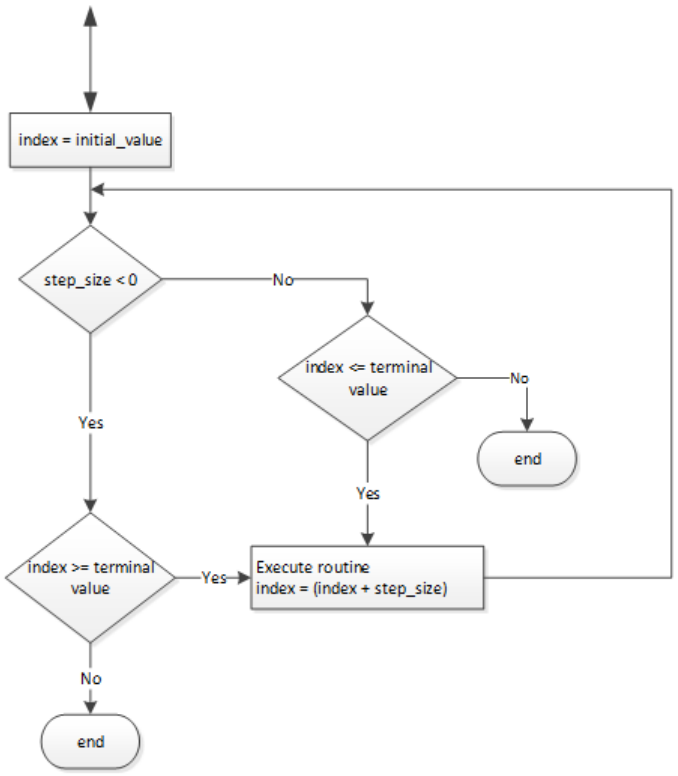
컨트롤러	메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	중첩 레벨 제한 > 25	4	94
	서브루틴은 SFC 이고 이미 실행 중입니다(재귀적인 호출)	4	82
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	N/A	N/A	N/A

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

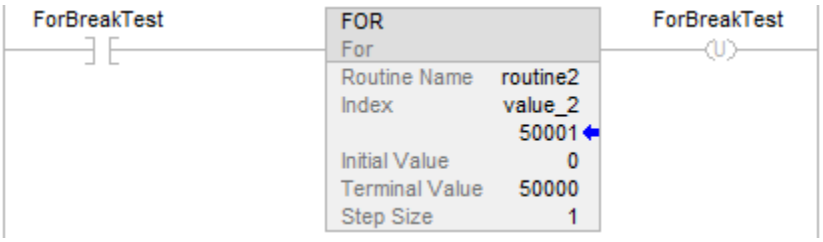
조건/상태	동작
사전 스캔	이전에 사전 스캔된 적이 없는 경우 명명된 서브루틴을 명령어가 사전 스캔합니다. **팁:** 재귀 FOR 명령어가 동일한 서브루틴에 있거나 여러 FOR 명령어가 동일한 서브루틴에 있을 경우(비재귀적), 서브루틴은 한 번만 사전 스캔됩니다. 하위가 JSR 에 의해 사전 스캔된 경우에도 마찬가지입니다.
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	다음 FOR 흐름도(참)를 참조하십시오.
사후 스캔	명령어는 명명된 서브루틴을 정확히 한 번 사후 스캔합니다.

FOR 흐름도(참)



예

활성화된 경우 FOR 명령어는 routine_2 를 반복적으로 실행하고 매번 value_2 를 1 씩 증가시킵니다. value_2 > 50000 이거나 BRK 명령어가 활성화된 경우 FOR 명령어는 더 이상 routine_2 를 실행하지 않습니다.



추가 참조

공통 속성 페이지의 963

서브루틴으로 이동(JSR), 서브루틴(SBR) 및 반환(RET)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

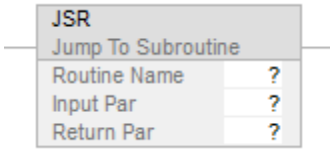
JSR 명령어는 다른 루틴을 호출합니다. 해당 루틴이 완료되면 실행이 JSR 명령어로 복귀합니다.

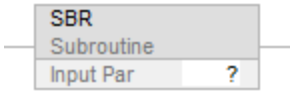
SBR 명령어는 JSR 에 의해 전달된 입력 파라미터를 수신합니다.

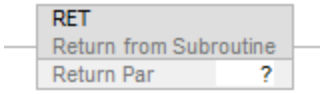
RET 명령어는 반환 파라미터를 JSR 에 다시 전달하고 서브루틴의 스캔을 종료합니다.

사용 가능한 언어

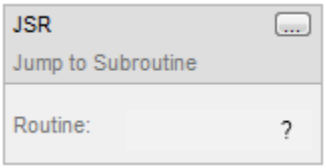
래더 다이어그램

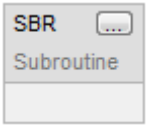


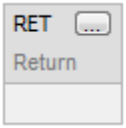




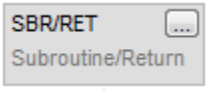
펑션 블록







순차 펑션 차트



ST(스트럭처드 텍스트)

JSR(RoutineName,InputCount,InputPar,ReturnPar);

SBR(InputPar);

RET(ReturnPar);

피연산자

중요: 다음과 같은 경우 작업 시 예외가 발생할 수 있습니다.

- 출력 태그 피연산자가 덮어써집니다.
- 구조 피연산자의 구성원이 덮어써집니다.
- 지정된 경우를 제외하고 여러 명령어에서 구조 피연산자를 공유합니다.

SBR 또는 RET 명령어의 각 파라미터에 JSR 명령어의 해당 파라미터와 같은 데이터 유형(모든 배열 차원 포함)을 사용합니다. 다른 데이터 유형을 사용하면 예기치 않은 결과가 발생할 수 있습니다.

래더 다이어그램

JSR 명령어

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
Routine Name	ROUTINE	ROUTINE	이름	실행할 서브루틴
Input Par	BOOL SINT INT DINT REAL 구조	BOOL SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL 구조	즉시 태그 배열 태그	이 루틴에서 서브루틴의 태그에 복사할 데이터. • 입력 파라미터는 옵션임 • 필요한 경우 입력 파라미터를 최대 40 개까지 입력합니다.

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
Return Par	BOOL SINT INT DINT REAL 구조	BOOL SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL 구조	태그 배열 태그	이 루틴에 서브루틴의 결과를 복사하도록 태그를 설정합니다. • 반환 파라미터는 옵션임 • 필요한 경우 반환 파라미터를 최대 40 개까지 입력

SBR 명령어

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
Input Par	BOOL SINT INT DINT REAL 구조	BOOL SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL 구조	태그 배열 태그	• 이 루틴의 태그이며, 이에 JSR 명령어의 해당 입력 파라미터(최대 40 개)를 복사.

RET 명령어

피연산자	데이터 유형 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	데이터 유형 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	형식	설명
Return Par	BOOL SINT INT DINT REAL 구조	BOOL SINT INT DINT LINT USINT UINT UDINT ULINT REAL LREAL 구조	즉시 태그 배열 태그	JSR 명령어의 해당 반환 파라미터(최대 40 개)에 복사할 이 루틴의 데이터.

연산 상태 플래그에 영향

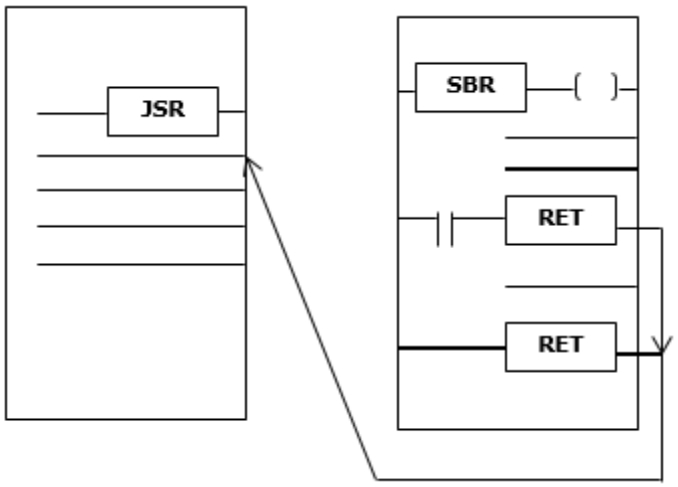
아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
JSR 명령어는 SBR 명령어보다 입력 파라미터 수가 적음	4	31
JSR 명령어가 폴트 루틴으로 이동	4	990 또는 사용자 제공
RET 명령어는 JSR 명령어보다 입력 파라미터 수가 적음	4	31
주 루틴에는 RET 명령어가 포함	4	31

연산

중요: 어떤 루틴이든 JSR 명령어를 포함할 수 있지만 JSR 명령어는 주 루틴을 호출(실행)할 수 없습니다.



JSR 명령어는 지정된 루틴(서브루틴이라 함)의 실행을 시작합니다.

- 서브루틴은 스캔될 때마다 실행합니다.
- 서브루틴이 실행한 후 로직 실행은 JSR 명령어를 포함한 루틴으로 돌아가서 JSR 다음의 명령어를 사용하여 계속합니다.

서브루틴에 대한 이동을 프로그래밍하려면 다음 지침을 따릅니다.

JSR

- 서브루틴의 태그에 데이터를 복사하려면 입력 파라미터를 입력합니다.
- 서브루틴의 결과를 이 루틴의 태그에 복사하려면 반환 파라미터를 입력합니다.
- 필요에 따라 입력을 40 개까지 입력하며 반환 파라미터를 40 개까지 입력합니다.

SBR

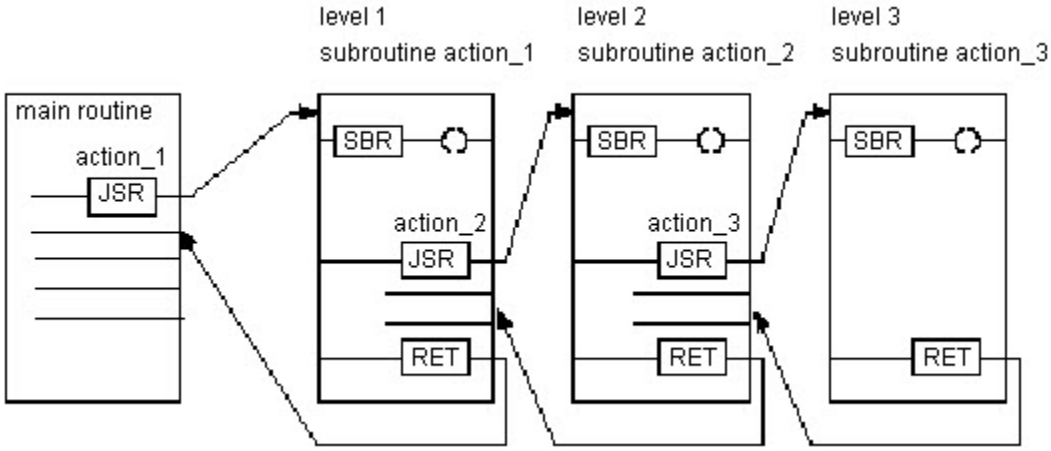
- JSR 명령어가 입력 파라미터를 포함한 경우 SBR 명령어를 입력합니다.

- SBR 명령어를 루틴의 첫 번째 명령어로 놓습니다.
- JSR 명령어의 각 입력 파라미터에 대해 데이터를 복사할 태그를 입력합니다.

RET

- JSR 명령어가 반환 파라미터를 포함한 경우 RET 명령어를 입력합니다.
- RET 명령어를 루틴의 마지막 명령어로 놓습니다.
- JSR 명령어의 각 반환 파라미터에 대해 JSR 명령어에 전달할 반환 파라미터를 입력합니다.
- 래더 루틴에서는 필요한 경우 다른 입력 조건을 기반으로 서브루틴을 종료하기 위해 추가 RET 명령어를 놓습니다(펑션 블록 루틴은 RET 명령어를 한 개만 허용).

최대 40 개의 파라미터를 서브루틴에 전달하고 최대 40 개의 파라미터가 서브루틴에서 반환되는 상태에서 중첩된 서브루틴을 25 개까지 호출합니다.



팁: 가변 피연산자를 추가 및 제거하려면 **편집 > 래더 요소 편집(Edit > Edit Ladder Element)** 메뉴를 선택합니다. JSR 및 SBR 명령어의 경우 입력 파라미터를 추가합니다. JSR 및 RET 명령어의 경우 출력 파라미터를 추가합니다. 3 개 명령어에 모두 명령어 파라미터를 제거합니다.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	렁은 거짓으로 설정됩니다. 컨트롤러가 모든 서브루틴을 실행합니다. 서브루틴의 모든 렁이 사전 스캔되도록 하기 위해 컨트롤러가 RET 명령어를 무시합니다(즉, RET 명령어가 서브루틴을 종료하지 않음). 입력 및 반환 파라미터는 전달되지 않습니다. 같은 서브루틴이 여러 번 호출되면 한 번만 사전 스캔됩니다.
렁-입력-조건이 (JSR 명령어에 대해) 거짓임	N/A
렁-입력-조건이 참임	파라미터가 전달되고 서브루틴이 실행됩니다.
사후 스캔	사전 스캔과 같은 조치

평션 블록

조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
EnableIn 이 거짓임	N/A
EnableIn 이 참임	파라미터가 전달되고 서브루틴이 실행됩니다.
명령어 최초 실행	N/A
명령어 최초 스캔	N/A
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 참조하십시오.

ST(스트럭처드 텍스트)

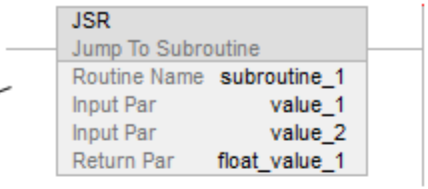
조건/상태	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	파라미터가 전달되고 서브루틴이 실행됩니다.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 참조하십시오.

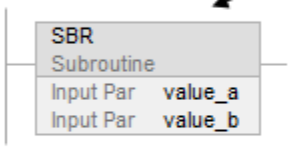
예제

예 1

래더 다이어그램

When enabled, the JSR instruction passes value_1 and value_2 to routine_1.

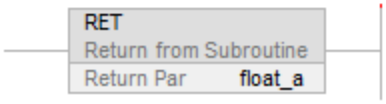




The SBR instruction receives value_1 and value_2 from the JSR instruction and copies those values to value_a and value_b, respectively. Logic execution continues in this routine.

[other rungs of code]

When enabled, the RET instruction sends float_a to the JSR instruction. The JSR instruction receives float_a and copies the value to float_value_1. Logic execution continues with the next instruction following the JSR instruction.



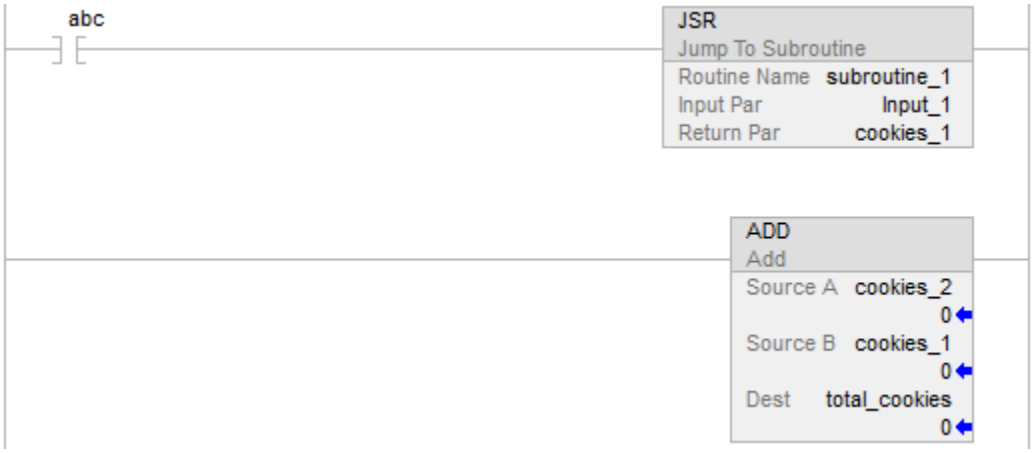
ST(스트럭처드 텍스트)

루틴	Program
주 루틴	JSR(routine_1,2,value_1,value_2,float_value_1);
서브루틴	SBR(value_a,value_b); <statements>; RET(float_a);

예 2

래더 다이어그램

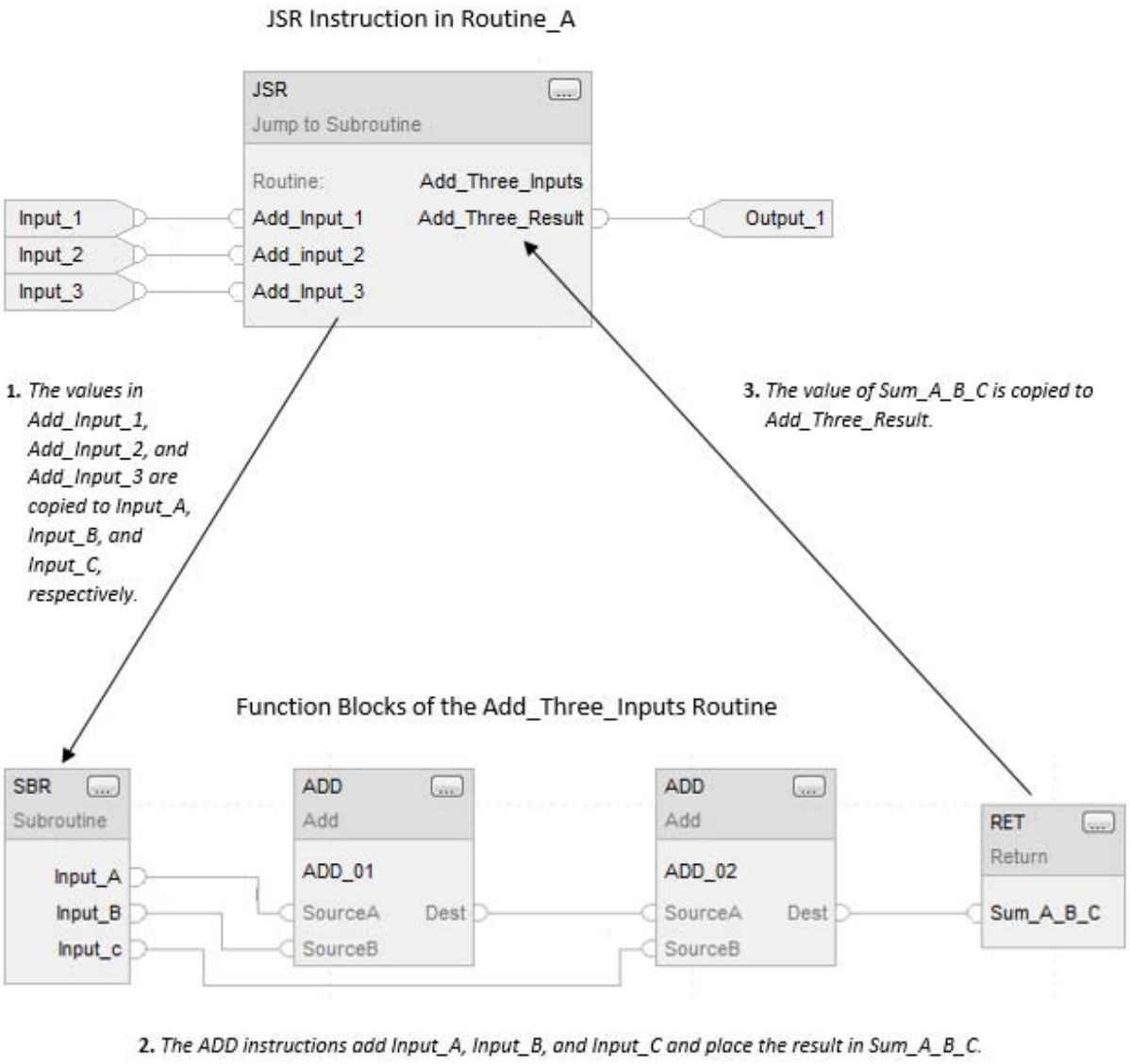
주 루틴



subroutine_1

예 3

펑션 블록



추가 참조

프로그램 제어 명령어 페이지의 680

배열을 통한 인덱스 페이지의 978

즉시 값 페이지의 967

특수 명령어

특수 명령어

특수 명령어는 응용 프로그램 특정 작업을 수행합니다.

사용 가능한 명령어

ST(스트럭처드 텍스트)

FBC	DDT	DTR	PID

평션 블록

사용할 수 없음

ST(스트럭처드 텍스트)

사용할 수 없음

실행할 작업:	사용할 명령어
데이터를 알려진 올바른 참조와 비교하고 불일치를 기록합니다.	FBC
데이터를 알려진 올바른 참조와 비교하고 불일치를 기록하며, 소스와 일치하도록 참조를 업데이트합니다.	DDT
마스크를 통해 소스 데이터를 전달하고 결과를 참조 데이터와 비교합니다. 다음에 비교하기 위해 참조에 소스를 씁니다.	DTR
PID 루프를 제어합니다.	PID

추가 참조

PID 명령어 사용 페이지의 775

수동에서 자동으로의 과적분 방지 및 무충돌 전환(PID) 페이지의 780

PID 명령어 타이밍 페이지의 784

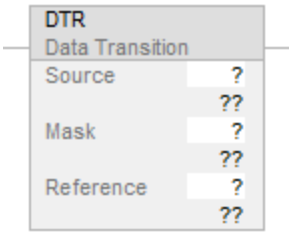
데이터 전환(DTR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

DTR 명령어는 Mask 를 통해 Source 값을 전달하고 결과를 Reference 값과 비교합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

래더 다이어그램

피연산자	유형	형식	설명
소스(Source)	DINT	즉시 태그	참조에 비교할 배열
마스크	DINT	즉시 태그	차단하거나 통과시킬 비트
참조	DINT	태그	소스에 비교할 배열

설명

DTR 명령어는 Mask 를 통해 Source 값을 전달하고 결과를 Reference 값과 비교합니다. 또한 DTR 명령어는 다음 비교를 위해 마스크된 Source 값을 Reference 값에 기록합니다. 소스는 변경되지 않습니다.

마스크의 "1"은 데이터 비트가 전달되었음을 의미합니다. 마스크의 "0"은 데이터 비트가 차단되었음을 의미합니다.

활성화된 경우 Mask 는 Mask 비트가 설정된 경우 데이터를 전달하며, Mask 비트가 해제된 경우 데이터를 차단합니다.

마스크된 Source 가 Reference 와 다르면 EnableOut 이 한 번 스캔에 대해 참입니다. 마스크된 Source 가 Reference 와 같으면 EnableOut 은 거짓입니다.

중요: 이 명령어를 이용한 온라인 프로그래밍은 위험할 수 있습니다. Reference 값이 Source 값과 다르면 EnableOut 은 참이 됩니다. 프로세서가 실행 또는 원격 실행 모드에 있을 때 이 명령어를 삽입하는 경우 주의해야 합니다.

즉시 마스크 값 입력

마스크를 입력하면 프로그래밍 소프트웨어가 기본적으로 십진수 값으로 설정됩니다. 다른 형식을 사용하여 마스크를 입력하려면 해당 값 앞에 올바른 접두어를 사용합니다.

접두사	설명
16#	16 진수(예: 16#0F0F)
8#	8 진수(예: 8#16)
2#	2 진수(예: 2#00110011)

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

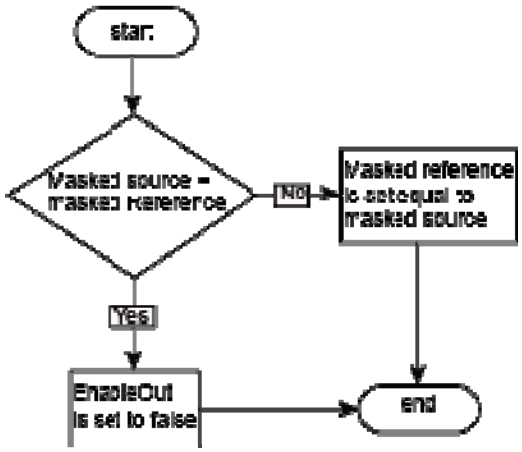
이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

래더 다이어그램

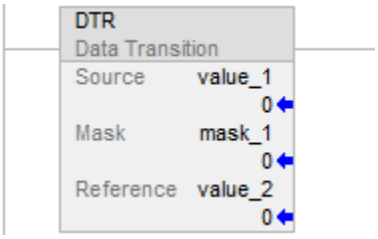
조건	동작
사전 스캔	Reference = Source AND Mask.
렁-입력-조건이 거짓임	Reference = Source AND Mask.
렁-입력-조건이 참임	DTR 흐름도(참)를 참조하십시오.
사후 스캔	N/A

DTR 흐름도(참)

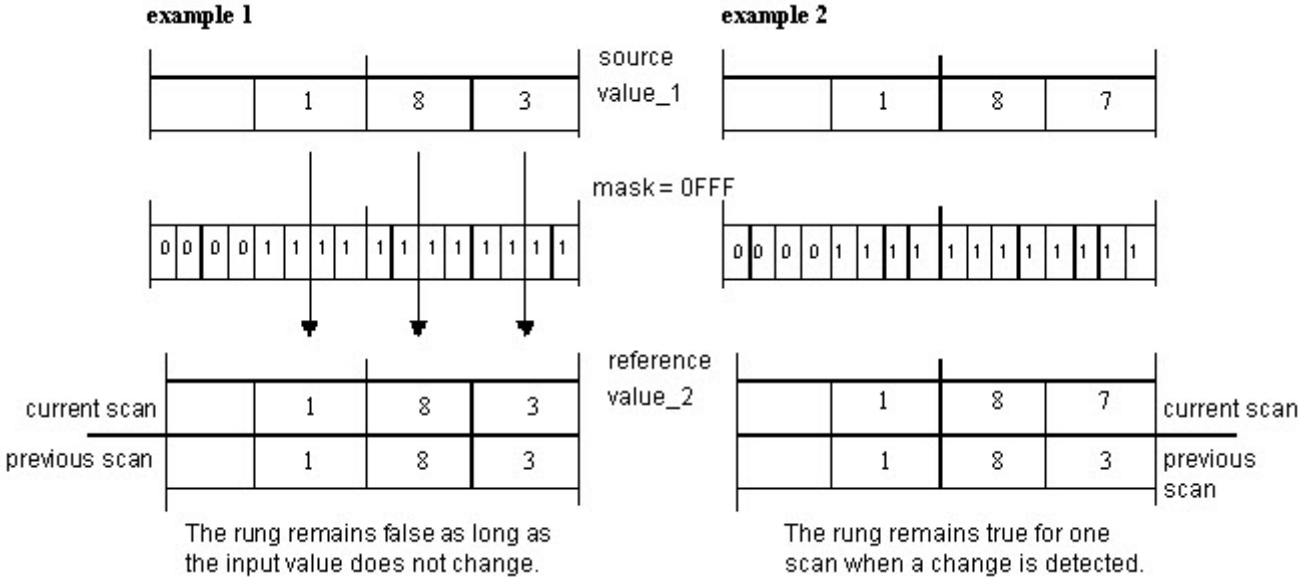


예

래더 다이어그램



활성화된 경우 DTR 명령어는 value_1 을 마스크합니다. 마스크한 두 값에 차이가 있으면 EnableOut 은 참으로 설정됩니다.



예 1 에서 기준값이 sourcevalue_1 AND mask 와 같으므로 EnableOut 은 항상 거짓으로 설정됩니다. 예 2 에서 어떤 이유로 소스 값이 변경되어 reference_value 가 source_value AND mask 와 다르므로 이 경우 EnableOut 은 참으로 설정되고 sourceValue 및 mask 를 기반으로 referencevalue 가 업데이트됩니다. 이전 스캔에서 기준값이 183 이지만 현재 스캔에서는 187 인 이유가 바로 이 때문입니다. 렁은 다음 스캔에서 소스가 변경되지 않는 한 기준값이 소스 값과 마스크 값의 논리곱 연산 결과와 다시 같아지게 되어 렁은 거짓으로 유지하므로 변경이 감지된 경우에 렁은 한 번 스캔에만 참으로 유지합니다.

추가 참조

특수 명령어 페이지의 745

FBC 페이지의 758

DDT 페이지의 749

공통 속성 페이지의 963

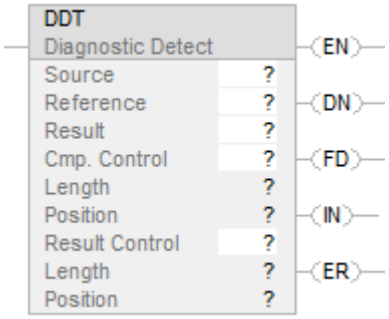
진단 감지(DDT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

DDT 명령어는 Source 배열의 비트를 Reference 배열의 비트와 비교하여 불일치 비트를 찾습니다. 그런 다음 불일치 비트 위치를 기록하고 불일치 Reference 비트를 Source 비트와 일치하도록 변경합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
소스	DINT	배열 태그	참조에 비교할 배열 첨자에 CONTROL.POS 를 사용하지 마십시오.
참조	DINT	배열 태그	소스에 비교할 배열 첨자에 CONTROL.POS 를 사용하지 마십시오.

Result	DINT	배열 태그	결과를 저장하는 배열 첨자에 CONTROL.POS 를 사용하지 마십시오.
Cmp. 제어	CONTROL	구조	비교의 제어 구조
길이(Length)	DINT	즉시	비교할 비트 수
위치(Position)	DINT	즉시	소스에서의 현재 위치 초기값은 일반적으로 0 입니다.
Result control	CONTROL	구조	결과의 제어 구조
길이(Length)	DINT	즉시	결과의 저장 위치 수
위치(Position)	DINT	즉시	결과에서의 현재 위치 초기값은 일반적으로 0 입니다.

중요: 비교 제어 구조 및 결과 제어 구조에 다른 태그를 사용합니다. 두 구조에 모두 같은 태그를 사용하면 예측할 수 없는 작업을 초래하여 장비 손상 및/또는 인원 부상을 야기할 수 있습니다.

COMPARE 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 DDT 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 DDT 명령어가 Source 및 Reference 배열의 마지막 비트를 비교할 때 설정됩니다.
.FD	BOOL	발견 비트는 DDT 명령어가 불일치를 발견할 때마다(한 번에 작업 한 개) 또는 모든 불일치를 기록한 후(스캔할 때 모두 작업) 설정됩니다.
.IN	BOOL	금지 비트는 DDT 검색 모드를 나타냅니다. 0 = 전체 모드 1 = 한 번에 불일치 한 개 모드
.ER	BOOL	에러 비트는 POS 나 LEN 이 유효하지 않은 경우 설정됩니다.
.LEN	DINT	길이 값은 비교할 비트 수를 식별합니다.
.POS	DINT	위치 값은 현재 비트를 식별합니다.

RESULT 구조

니모닉	데이터 유형	설명
.DN	BOOL	완료 비트는 Result 배열이 가득 찬 경우 설정됩니다.
.LEN	DINT	길이 값은 Result 배열의 저장 위치 수를 식별합니다.
.POS	DINT	위치 값은 Result 배열의 위치 수를 식별합니다.

설명

활성화된 경우 DDT 명령어는 Source 배열의 비트를 Reference 배열의 비트와 비교하고 각 불일치 비트 수를 Result 배열에 기록하고 Reference 비트의 값을 해당 Source 의 값과 일치하도록 변경합니다.

중요: DDT 명령어는 연속 메모리에서 작동됩니다. 명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

DDT 와 FBC 명령어 간의 차이점은 DDT 명령어가 불일치를 발견할 때마다 참조 비트를 소스 비트와 일치하도록 변경한다는 것입니다. FBC 명령어는 참조 비트를 변경하지 않습니다.

이 명령어가 배열의 끝을 지나서 읽기 작업을 수행하려고 할 경우 .ER 비트를 설정하여 메이저 폴트가 발생합니다.

검색 모드 선택

감지할 항목:	선택할 모드:
한 번에 불일치 한 개	비교 CONTROL 구조에서 .IN 비트를 설정합니다. EnableIn 이 거짓에서 참으로 전환할 때마다 DDT 명령어는 Source 와 Reference 배열 간의 다음 불일치를 검색합니다. 불일치를 찾으면 명령어가 정지하고 .FD 비트를 설정하며 불일치의 위치를 기록합니다.
모든 불일치	비교 CONTROL 구조의 .IN 비트를 해제합니다. EnableIn 이 거짓에서 참으로 전환할 때마다 DDT 명령어는 Source 와 Reference 배열 간의 모든 불일치를 검색합니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
result.POS > 결과 배열의 크기	4	20

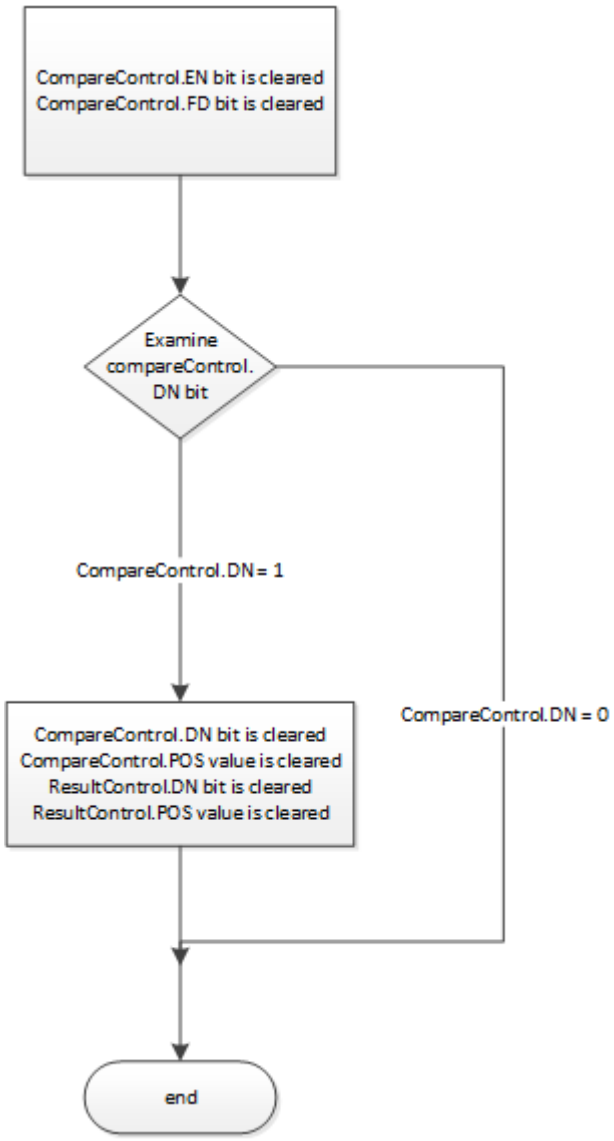
피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

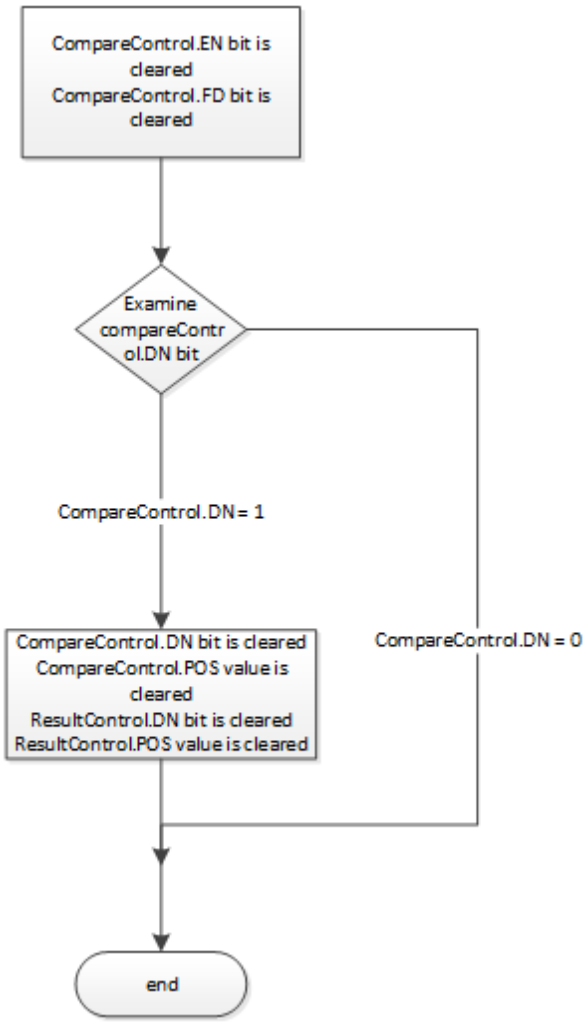
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	DDT 흐름도(사전 스캔)를 참조하십시오.
렁-입력-조건이 거짓임	DDT 흐름도(거짓)를 참조하십시오.
렁-입력-조건이 참임	DDT 흐름도(참)를 참조하십시오.
사후 스캔	N/A

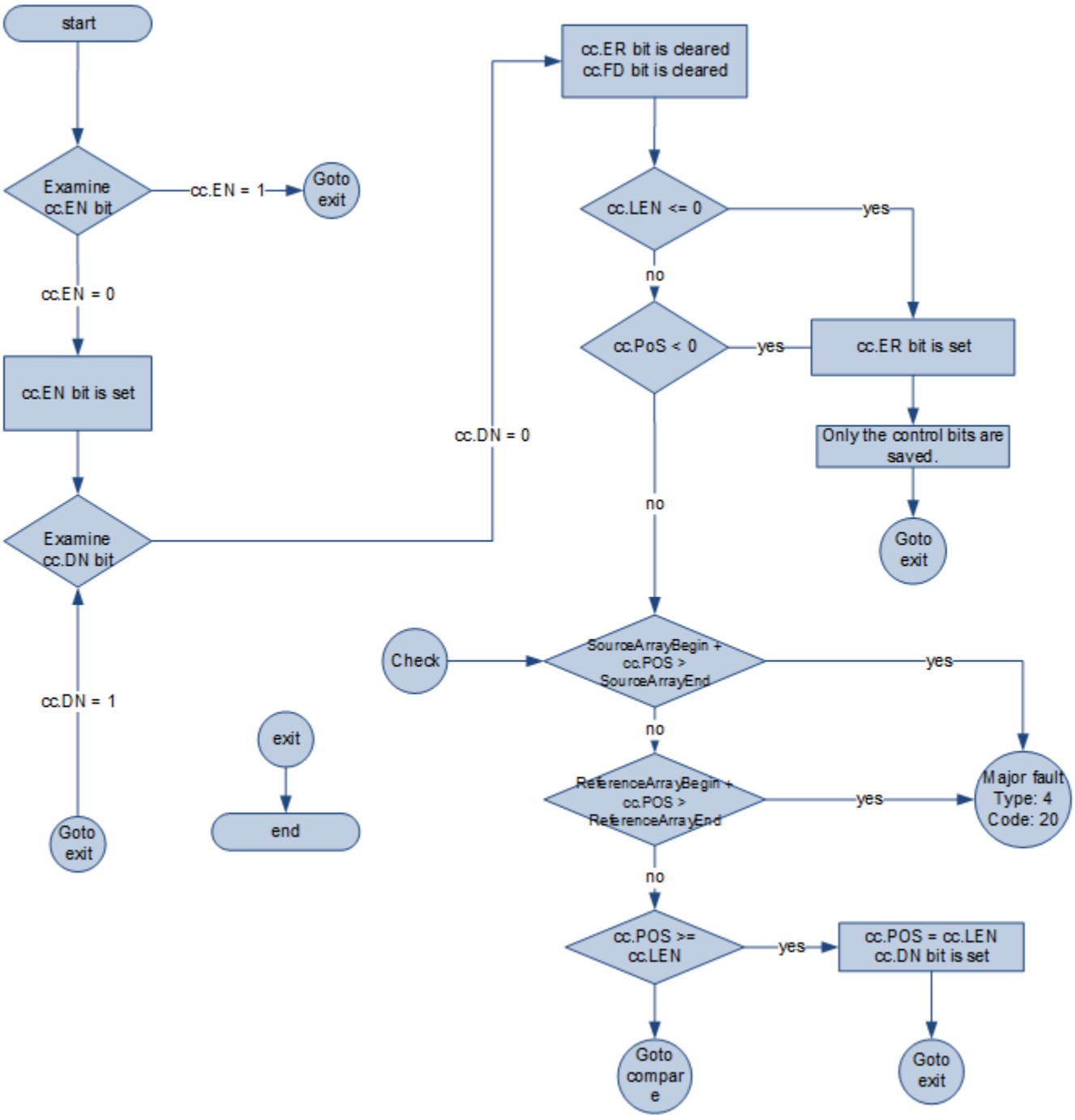
DDT 흐름도(사전 스캔)


CompareControl.EN bit is cleared
CompareControl.FD bit is cleared
Examine compareControl. DN bit
CompareControl.DN = 1
CompareControl.DN = 0
CompareControl.DN bit is cleared
CompareControl.POS value is cleared
ResultControl.DN bit is cleared
ResultControl.POS value is cleared
end

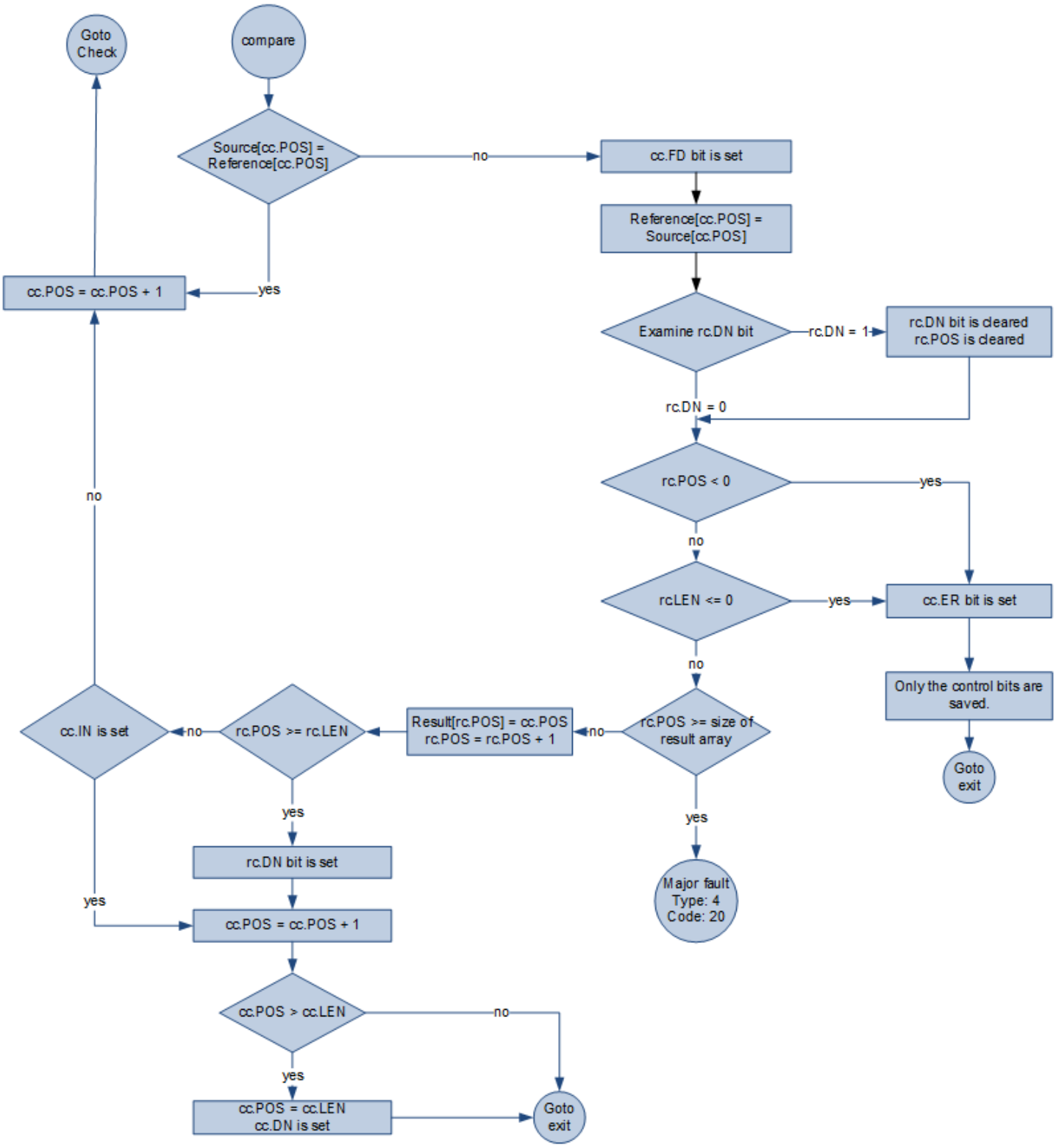
DDT 흐름도(거짓)


CompareControl.EN bit is cleared
CompareControl.FD bit is cleared
Examine compareControl.DN bit
CompareControl.DN = 1
CompareControl.DN = 0
CompareControl.DN bit is cleared
CompareControl.POS value is cleared
ResultControl.DN bit is cleared
ResultControl.POS value is cleared
end

DDT 흐름도(참)

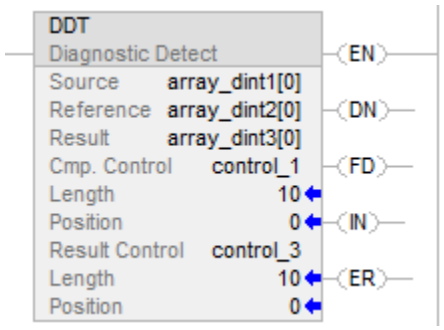

start
Examine cc.EN bit
cc.EN = 1
Goto exit
cc.EN = 0
cc.EN bit is set
Examine cc.DN bit
cc.DN = 0
cc.DN = 1
Goto exit
exit
end
cc.ER bit is cleared
cc.FD bit is cleared
cc.LEN <= 0
yes
no
cc.PoS < 0
yes
cc.ER bit is set
Only the control bits are saved.
Goto exit
no
Check
SourceArrayBegin + cc.POS > SourceArrayEnd
yes
no
ReferenceArrayBegin + cc.POS > ReferenceArrayEnd
yes
Major fault Type: 4 Code: 20
no
cc.POS >= cc.LEN
yes
cc.POS = cc.LEN cc.DN bit is set
Goto compare
Goto exit

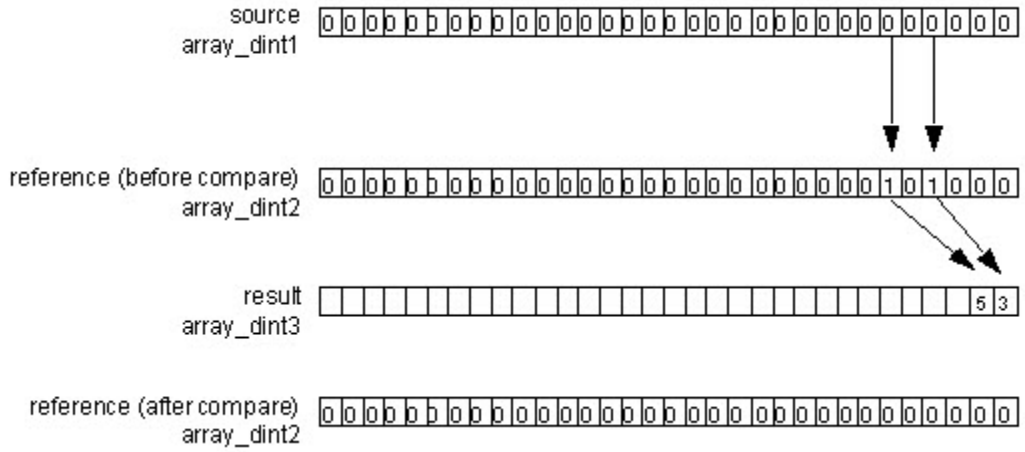
DDT 흐름도(참) – 계속


Goto Check
compare
Source[cc.POS] = Reference[cc.POS]
no
cc.FD bit is set
Reference[cc.POS] = Source[cc.POS]
cc.POS = cc.POS + 1
yes
Examine rc.DN bit
rc.DN = 1
rc.DN bit is cleared
rc.POS is cleared
rc.DN = 0
rc.POS < 0
yes
no
rc.LEN <= 0
yes
cc.ER bit is set
no
Only the control bits are saved.
Goto exit
rc.POS >= size of result array
no
Result[rc.POS] = cc.POS
rc.POS = rc.POS + 1
rc.POS >= rc.LEN
no
cc.IN is set
no
yes
rc.DN bit is set
yes
cc.POS = cc.POS + 1
Major fault
Type: 4
Code: 20
cc.POS > cc.LEN
no
yes
cc.POS = cc.LEN
cc.DN is set
Goto exit

예제

래더 다이어그램





추가 참조

특수 명령어 페이지의 745

DTR 페이지의 746

FBC 페이지의 758

공통 특성 페이지의 963

데이터 변환 페이지의 967

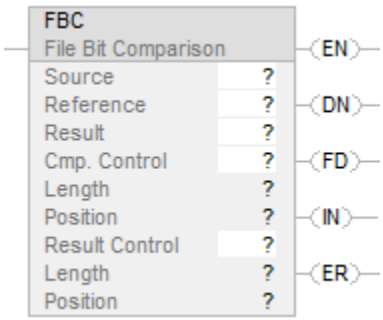
파일 비트 비교(FBC)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

FBC 명령어는 Source 배열의 비트를 Reference 배열의 비트와 비교합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
소스	DINT	배열 태그	참조에 비교할 배열 첨자에 CONTROL.POS 를 사용하지 **마십시오**.
참조	DINT	배열 태그	소스에 비교할 배열 첨자에 CONTROL.POS 를 사용하지 마십시오.
Result	DINT	배열 태그	결과를 저장하는 배열 첨자에 CONTROL.POS 를 사용하지 마십시오.
Cmp. 제어	CONTROL	구조	비교의 제어 구조

길이(Length)	DINT	즉시	비교할 비트 수
위치(Position)	DINT	즉시	소스에서의 현재 위치 초기값은 일반적으로 0 입니다.
Result control	CONTROL	구조	결과의 제어 구조
길이(Length)	DINT	즉시	결과의 저장 위치 수
위치(Position)	DINT	즉시	결과에서의 현재 위치 초기값은 일반적으로 0 입니다.

중요: 비교 제어 구조 및 결과 제어 구조에 다른 태그를 사용합니다. 두 구조에 모두 같은 태그를 사용하면 예측할 수 없는 작업을 초래하여 장비 손상 및/또는 인원 부상을 야기할 수 있습니다.

COMPARE 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 FBC 명령어가 활성화되었음을 나타냅니다.
.DN	BOOL	완료 비트는 FBC 명령어가 Source 및 Reference 배열의 마지막 비트를 비교할 때 설정됩니다.
.FD	BOOL	발견 비트는 FBC 명령어가 불일치를 기록할 때마다(한 번에 작업 한 개) 또는 모든 불일치를 기록한 후(스캔할 때 모두 작업) 설정됩니다.
.IN	BOOL	금지 비트는 FBC 검색 모드를 나타냅니다. 0 = 전체 모드 1 = 한 번에 불일치 한 개 모드
.ER	BOOL	에러 비트는 POS 나 LEN 이 유효하지 않으면 설정됩니다.
.LEN	DINT	길이 값은 비교할 비트 수를 식별합니다.
.POS	DINT	위치 값은 현재 비트를 식별합니다.

RESULT 구조

니모닉	데이터 유형	설명
.DN	BOOL	완료 비트는 Result 배열이 가득 찬 경우 설정됩니다.
.LEN	DINT	길이 값은 Result 배열의 저장 위치 수를 식별합니다.
.POS	DINT	위치 값은 Result 배열의 위치 수를 식별합니다.

설명

활성화된 경우 FBC 명령어는 Source 배열의 비트를 Reference 배열의 비트와 비교하고 각 불일치 비트 수를 Result 배열에 기록합니다.

중요: FBC 명령어는 연속 메모리에서 작동됩니다. 명령어가 변경하지 않으려는 데이터를 변경하지 않는지 테스트하고 확인해야 합니다.

DDT 와 FBC 명령어 간의 차이점은 DDT 명령어가 불일치를 발견할 때마다 참조 비트를 소스 비트와 일치하도록 변경한다는 것입니다. FBC 명령어는 참조 비트를 변경하지 않습니다.

이 명령어가 배열의 끝을 지나서 읽기 작업을 수행하려고 할 경우 .ER 비트를 설정하여 메이저 폴트가 발생합니다.

검색 모드 선택

감지할 항목:	선택할 모드:
한 번에 불일치 한 개	비교 CONTROL 구조에서 .IN 비트를 설정합니다. EnableIn 이 거짓에서 참으로 전환할 때마다 FBC 명령어는 Source 와 Reference 배열 간의 다음 불일치를 검색합니다. 불일치를 찾으면 이 명령어는 .FD 비트를 설정하고 불일치의 위치를 기록하고 실행을 정지합니다.
모든 불일치	비교 CONTROL 구조의 .IN 비트를 해제합니다. EnableIn 이 거짓에서 참으로 전환할 때마다 FBC 명령어는 Source 와 Reference 배열 간의 모든 불일치를 검색합니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
result.POS > 결과 배열의 크기	4	20

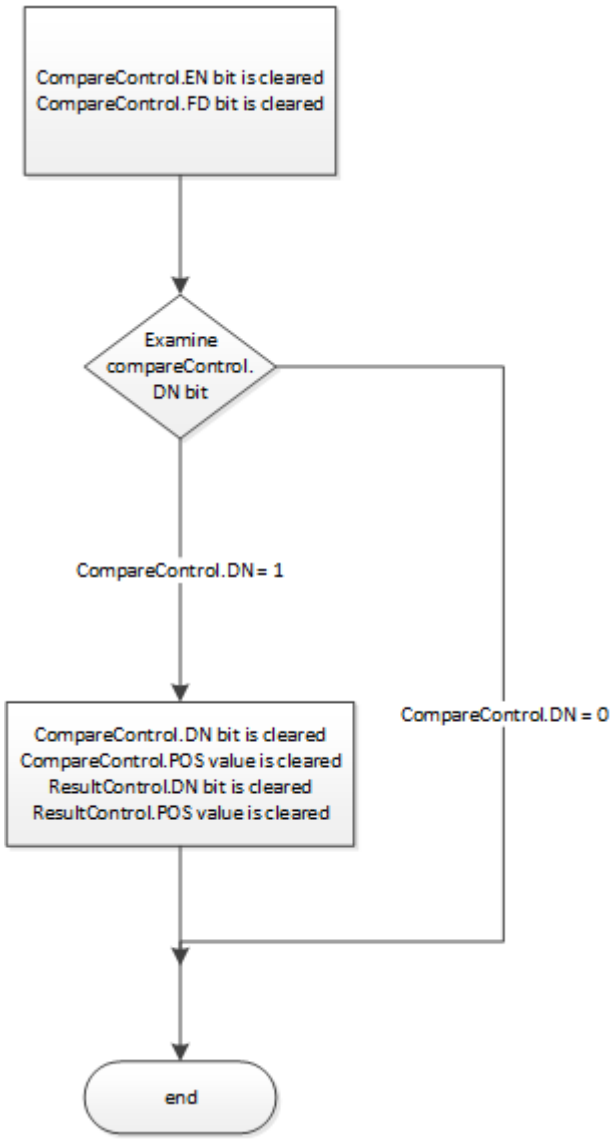
피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

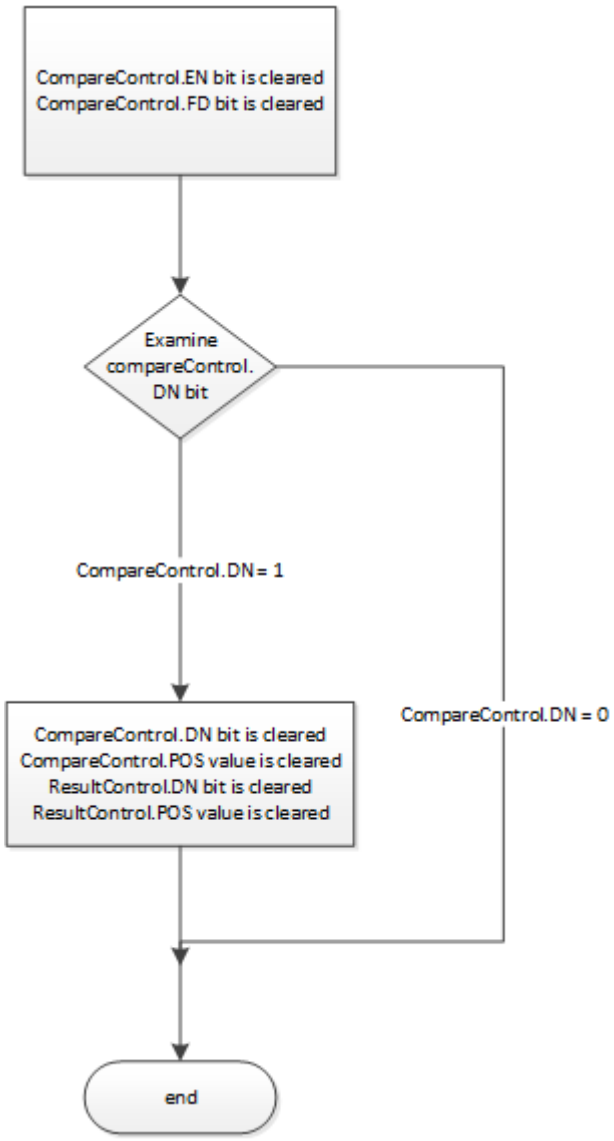
래더 다이어그램

조건/상태	취해진 조치
사전 스캔	FBC 흐름도(사전 스캔)를 참조하십시오.
렁-입력-조건이 거짓임	FBC 흐름도(거짓)를 참조하십시오.
렁-입력-조건이 참임	FBC 흐름도(참)를 참조하십시오.
사후 스캔	N/A

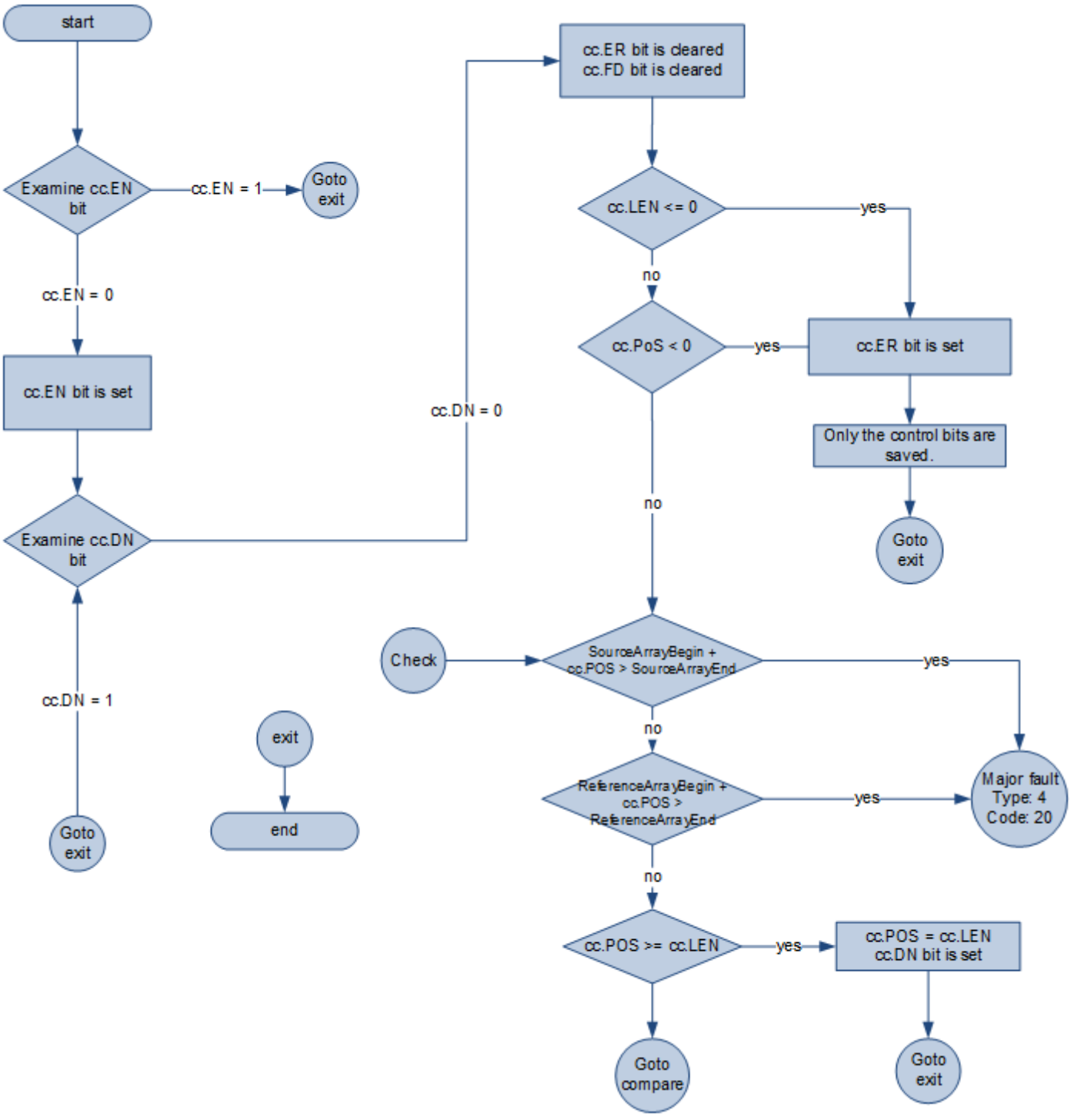
FBC 흐름도(사전 스캔)


CompareControl.EN bit is cleared
CompareControl.FD bit is cleared
Examine compareControl. DN bit
CompareControl.DN = 1
CompareControl.DN = 0
CompareControl.DN bit is cleared
CompareControl.POS value is cleared
ResultControl.DN bit is cleared
ResultControl.POS value is cleared
end

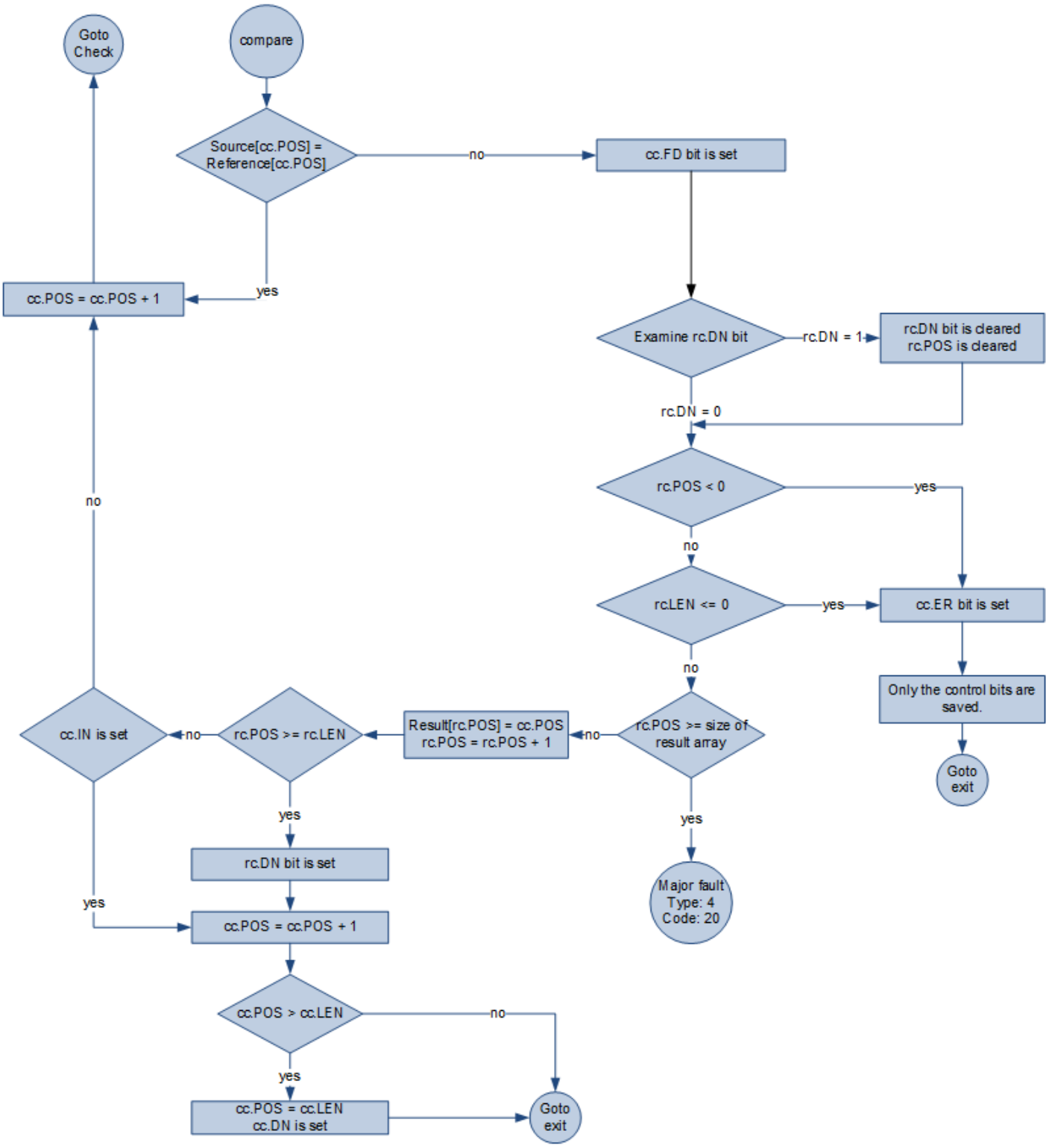
FBC 흐름도(거짓)


CompareControl.EN bit is cleared
CompareControl.FD bit is cleared
Examine compareControl. DN bit
CompareControl.DN = 1
CompareControl.DN = 0
CompareControl.DN bit is cleared
CompareControl.POS value is cleared
ResultControl.DN bit is cleared
ResultControl.POS value is cleared
end

FBC 흐름도(참)

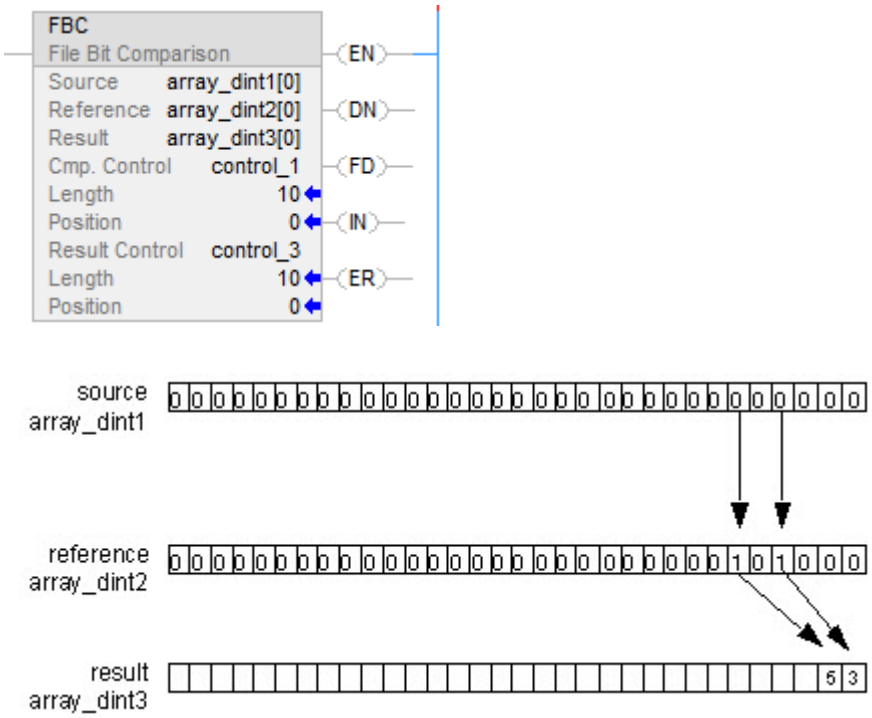

start
Examine cc.EN bit
cc.EN = 1
Goto exit
cc.EN = 0
cc.EN bit is set
Examine cc.DN bit
cc.DN = 0
cc.DN = 1
Goto exit
exit
end
cc.ER bit is cleared
cc.FD bit is cleared
cc.LEN <= 0
yes
no
cc.PoS < 0
yes
cc.ER bit is set
Only the control bits are saved.
Goto exit
no
Check
SourceArrayBegin + cc.POS > SourceArrayEnd
yes
no
ReferenceArrayBegin + cc.POS > ReferenceArrayEnd
yes
Major fault Type: 4 Code: 20
no
cc.POS >= cc.LEN
yes
cc.POS = cc.LEN
cc.DN bit is set
Goto compare
Goto exit

FBC 흐름도(참) - 계속


Goto Check
compare
Source[cc.POS] = Reference[cc.POS]
no
cc.FD bit is set
yes
cc.POS = cc.POS + 1
Examine rc.DN bit
rc.DN = 1
rc.DN bit is cleared
rc.POS is cleared
rc.DN = 0
rc.POS < 0
yes
no
rc.LEN <= 0
yes
cc.ER bit is set
Only the control bits are saved.
Goto exit
no
no
cc.IN is set
no
rc.POS >= rc.LEN
Result[rc.POS] = cc.POS
rc.POS = rc.POS + 1
no
rc.POS >= size of result array
yes
yes
rc.DN bit is set
Major fault
Type: 4
Code: 20
yes
cc.POS = cc.POS + 1
cc.POS > cc.LEN
no
yes
cc.POS = cc.LEN
cc.DN is set
Goto exit

예

래더 다이어그램



추가 참조

특수 명령어 페이지의 745

DDT 페이지의 749

DTR 페이지의 746

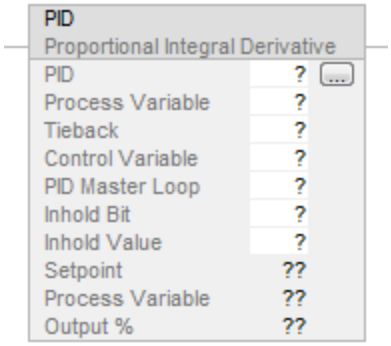
비례-적분-미분(PID)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

PID 명령어는 흐름, 압력, 온도 또는 레벨 등과 같은 프로세스 변수를 제어합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

PID(PID,ProcessVariable,Tieback,ControlVariable,PIDMasterLoop,InHoldBit,InHoldValue);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
PID	PID	구조	PID 구조
Process variable	SINT	태그	제어할 값
	INT		
	DINT		
	REAL		
Tieback	SINT	immediate	(옵션)
	INT	태그	
	DINT		컨트롤러의 출력을 무시하는 하드웨어 수동/자동 스테이션의 출력. 이 파라미터를 사용하지 않으려면 0 을 입력합니다.

	REAL		
Control variable	SINT	태그	최종 제어 장치(밸브, 댐퍼 등)로 보내는 값
	INT		
	DINT		불감대를 사용하는 경우, Control variable 은 REAL 이어야 하며 에러가 불감대 내에 있을 경우 강제로 0 이 됩니다.
	REAL		
PID master loop	PID	Structure	옵션
			마스터 PID 에 대한 PID 태그
			캐스케이드 제어를 수행하는데 이 PID 가 슬레이브 루프인 경우 마스터 PID 의 이름을 입력합니다.
			이 파라미터를 사용하지 않으려면 0 을 입력합니다.
Inhold bit	BOOL	태그	옵션
			1756 아날로그에서 오는 인홀드 비트의 현재 상태
			무충돌 다시 시작을 지원하는 출력 채널
Inhold value	SINT	태그	옵션
	INT		1756 아날로그 출력에서 오는 데이터 리드백 값
	DINT		무충돌 다시 시작을 지원하는 채널
	REAL		이 파라미터를 사용하지 않으려면 0 을 입력합니다.
Setpoint			표시 전용
			설정값의 현재 값
Process variable			표시 전용
			스케일링된 Process_Variable 의 현재 값
Output %			표시 전용
			현재 출력 백분율 값

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
PID	PID	구조	PID 구조
Process variable	SINT	태그	제어할 값
	INT		
	DINT		
	REAL		
Tieback	SINT	immediate	(옵션)
	INT	태그	
	DINT		컨트롤러의 출력을 무시하는 하드웨어 수동/자동 스테이션의 출력. 이 파라미터를 사용하지 않으려면 0 을 입력합니다.
	REAL		
Control variable	SINT	태그	최종 제어 장치(밸브, 댐퍼 등)로 보내는 값
	INT		
	DINT		불감대를 사용하는 경우, Control variable 은 REAL 이어야 하며 에러가 불감대 내에 있을 경우 강제로 0 이 됩니다.
	REAL		
PID master loop	PID	Structure	옵션
			마스터 PID 에 대한 PID 태그
			캐스케이드 제어를 수행하는데 이 PID 가 슬레이브 루프인 경우 마스터 PID 의 이름을 입력합니다.
			이 파라미터를 사용하지 않으려면 0 을 입력합니다.
Inhold bit	BOOL	태그	옵션
			1756 아날로그에서 오는 인홀드 비트의 현재 상태
			무충돌 다시 시작을 지원하는 출력 채널
Inhold value	SINT	태그	옵션

	INT		1756 아날로그 출력에서 오는 데이터 리드백 값
	DINT		무충돌 다시 시작을 지원하는 채널
	REAL		이 파라미터를 사용하지 않으려면 0 을 입력합니다.
Setpoint			표시 전용
			설정값의 현재 값
Process variable			표시 전용
			스케일링된 Process_Variable 의 현재 값
Output %			표시 전용
			현재 출력 백분율 값

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

PID 구조

각 PID 명령어에 대해 고유한 PID 구조를 지정합니다.

니모닉	데이터 유형	설명		
.CTL	DINT	.CTL 구성원은 32 비트 워드 한 개에서 상태 구성원(비트)에 대한 액세스를 제공합니다. 비트 07 ~ 15 는 PID 명령어가 설정합니다.		
		비트	**번호**	**설명**
		.EN	31	
		.CT	30	캐스케이드 유형(0 = 슬레이브; 1 = 마스터)
		.CL	29	캐스케이드 루프(0 = 아니오; 1 = 예)
		.PVT	28	프로세스 변수 추적(0 = 아니오; 1 = 예)
		.DOE	27	(0 = PV; 1 = 에러)의 편차
		.SWM	26	소프트웨어 모드(0= 아니오 - 자동; 1 = 예- 소프트웨어 수동)
		.CA	25	제어 동작(0 = 역방향(SP-PV); 1 = 직접(PV-SP))

니모닉	데이터 유형	설명		
		.MO	24	스테이션 모드(0 = 자동; 1 = 수동)
		.PE	23	PID 수식(0 = 독립; 1 = 종속)
		.NDF	22	미분 평탄화(0 = 아니오; 1 = 예)
		.NOBC	21	편차 계산(0 = 아니오; 1 = 예)
		.NOZC	20	0 점 교차(0 = 아니오; 1 = 불감대의 경우)
		.INI	15	PID 초기화됨(0 = 아니오; 1 = 예)
		.SPOR	14	설정값 범위 초과(0 = 아니오; 1 = 예)
		.OLL	13	CV 가 최소 출력 값보다 낮음(0 = 아니오; 1 = 예)
		.OLH	12	CV 가 최대 출력 값보다 높음(0 = 아니오; 1 = 예)
		.EWD	11	에러가 불감대 이내(0 = 아니오; 1 = 예)
		.DVNA	10	에러 하한 알람(0 = 아니오; 1 = 예)
		.DVPA	9	에러 상한 알람(0 = 아니오; 1 = 예)
		.PVLA	8	PV 가 하한 알람(0 = 아니오; 1 = 예)
		.PVHA	7	PV 가 상한 알람(0 = 아니오; 1 = 예)
.SP	REAL	설정값		
.KP	REAL	독립 - 비례 게인(단위 없음)		
		종속 - 컨트롤러 게인(단위 없음)		
.KI	REAL	독립 - 적분 게인(1/초)		
		종속 - 리셋 시간(반복당 분)		
.KD	REAL	독립 - 미분 게인(초)		
		종속 - 미분 시간(분)		
.BIAS	REAL	피드포워드 또는 편차 %		
.MAXS	REAL	최대 엔지니어링 단위 스케일링 값		

니모닉	데이터 유형	설명
.MINS	REAL	최소 엔지니어링 단위 스케일링 값
.DB	REAL	불감대 엔지니어링 단위
.SO	REAL	출력 % 설정
.MAXO	REAL	최대 출력 제한(출력의 %)
.MINO	REAL	최소 출력 제한(출력의 %)
.UPD	REAL	루프 업데이트 시간(초)
.PV	REAL	스케일링된 PV 값
.ERR	REAL	스케일링된 에러 값
.OUT	REAL	출력 %
.PVH	REAL	프로세스 변수 상한 알람 제한
.PVL	REAL	프로세스 변수 하한 알람 제한
.DVP	REAL	양수 편차 알람 제한
.DVN	REAL	음수 편차 알람 제한
.PVDB	REAL	프로세스 변수 알람 불감대
.DVDB	REAL	편차 알람 불감대
.MAXI	REAL	최대 PV 값(스케일링되지 않은 입력)
.MINI	REAL	최소 PV 값(스케일링되지 않은 입력)
.TIE	REAL	수동 제어에 대한 타이백 값
.MAXCV	REAL	최대 CV 값(100%에 해당)
.MINCV	REAL	최소 CV 값(0%에 해당)
.MINTIE	REAL	최소 타이백 값(100%에 해당)
.MAXTIE	REAL	최대 타이백 값(0%에 해당)
.DATA[17]	REAL	**.DATA 구성원 저장소:**

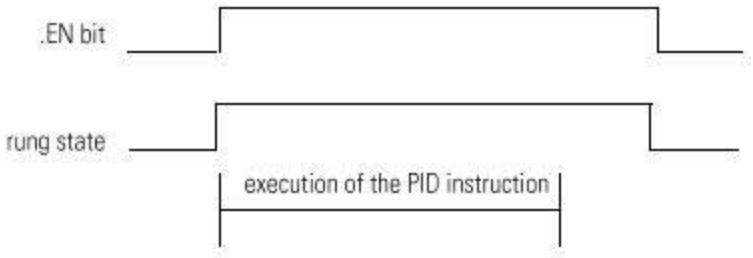
요소	설명
.DATA[0]	적분항 누산
.DATA[1]	미분 평탄 임시 값
.DATA[2]	이전 .PV 값
.DATA[3]	이전 .ERR 값
.DATA[4]	이전 유효한 .SP 값
.DATA[5]	백분율 스케일링 상수

니모닉	데이터 유형	설명	
		.DATA[6]	.PV 스케일링 상수
		.DATA[7]	미분 스케일링 상수
		.DATA[8]	이전 .KP 값
		.DATA[9]	이전 .KI 값
		.DATA[10]	이전 .KD 값
		.DATA[11]	종속 게인 .KP
		.DATA[12]	종속 게인 .KI
		.DATA[13]	종속 게인 .KD
		.DATA[14]	이전 .CV 값
		.DATA[15]	.CV 스케일링 해제 상수
		.DATA[16]	타이백 스케일링 해제 상수

설명

PID 명령어는 일반적으로 프로세스 변수를 원하는 설정값으로 유지하기 위해 아날로그 모듈에서 프로세스 변수(PV)를 받으며 아날로그 출력 모듈의 제어 변수 출력(CV)을 조절합니다.

.EN 비트는 실행 상태를 나타냅니다. .EN 비트는 EnableIn 이 거짓에서 참으로 전환할 때 설정됩니다. .EN 비트는 EnableIn 이 거짓이 될 때 해제됩니다. PID 명령어는 .DN 비트를 사용하지 않습니다. PID 명령어는 EnableIn 가 참인 한 모든 스캔을 실행합니다.



연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

마이너 폴트 발생 조건:	폴트 유형	폴트 코드
UPD ≥ 0	4	35
설정값 범위 초과	4	36

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

추가 참조

특수 명령어 페이지의 745

공통 특성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

PID 명령어 사용

PID 명령어를 입력하고 PID 구조를 지정한 후 구성 탭을 사용하여 PID 명령어의 작동 방식을 지정합니다.

튜닝 지정

튜닝(Tuning) 탭을 선택합니다. 다른 필드, **확인**(OK) 또는 **적용**(Apply)을 클릭하거나 **Enter** 키를 누르면 변경 내용이 즉시 적용됩니다.

해당 필드:	다음을 수행합니다.
설정값(Setpoint)(SP)	설정값(.SP)을 입력합니다.
출력 % 설정(Set output %)	출력 퍼센트 설정(.SO)를 입력합니다. 소프트웨어 수동 모드의 출력에 이 값이 사용됩니다. 자동 모드에서 이 값은 출력 %를 표시합니다.
출력 편차(Output bias)	출력 편차 퍼센트(.BIAS)를 입력합니다.
비례 게인(Kp)(Proportional gain (Kp))	비례 게인(.KP)을 입력합니다. 독립 게인의 경우 비례 게인입니다(단위 없음). 종속 게인의 경우 컨트롤러 게인입니다(단위 없음).
적분 게인(Ki)(Integral gain (Ki))	적분 게인(.KI)을 입력합니다. 독립 게인의 경우 적분 게인입니다(1/초). 종속 게인의 경우 리셋 시간입니다(반복당 분).

미분 시간(Kd)(Derivative time (Kd))	미분 게인(.KD)을 입력합니다. 독립 게인의 경우 미분 게인(초)입니다. 종속 게인의 경우 비율 시간(분)입니다.
수동 모드(Manual mode)	수동(.MO) 또는 소프트웨어 수동(.SWM)을 선택합니다. 수동 모드와 소프트웨어 수동 모드를 모두 선택하면 수동 모드가 우선합니다.

구성 지정

구성(Configuration) 탭을 선택합니다. 변경 내용은 확인(OK) 또는 적용(Apply)을 클릭해야 적용됩니다.

해당 필드:	**다음을 수행합니다.**
PID 수식(PID equation)	독립 게인 또는 종속 게인(.PE)을 선택합니다. 3 가지 게인(P, I 및 D)을 독립적으로 작동하게 하려면 독립을 사용합니다. 3 개의 항(P, I 및 D) 모두에 영향을 미치는 전체 컨트롤러 게인이 필요하면 종속을 사용합니다.
제어 동작(Control action)	제어 동작(.CA)에 대해 E=PV-SP 또는 E=SP-PV 를 선택합니다.
미분 대상(Derivative of)	PV 또는 에러(.DOE)를 선택합니다. 설정값을 변경하면 생기는 출력 스파이크 위험을 줄이려면 PV 의 미분을 사용합니다. 알고리즘에서 오버슈트를 허용할 때 설정값 변경에 대한 빠른 응답을 얻으려면 에러의 미분을 사용합니다.
루프 업데이트 시간(Loop update time)	명령에 대한 업데이트 시간(.UPD)을 입력합니다.
CV 상한(CV high limit)	제어 변수에 대한 상한(.MAXO)을 입력합니다.(1)
CV 하한(CV low limit)	제어 변수에 대한 하한(.MINO)을 입력합니다.(1)
불감대 값(Deadband value)	불감대 값(.DB)을 입력합니다.
미분 평탄화 없음(No derivative smoothing)	이 선택 항목(.NDF)을 활성화 또는 비활성화합니다.
편차 계산 없음(No bias calculation)	이 선택 항목(.NOBC)을 활성화 또는 비활성화합니다.
불감대에 0 점 교차 없음	이 선택 항목(.NOZC)을 활성화 또는 비활성화합니다.
PV 추적(PV tracking)	이 선택 항목(.PVT)을 활성화 또는 비활성화합니다.
캐스케이드 루프(Cascade loop)	이 선택 항목(.CL)을 활성화 또는 비활성화합니다.
캐스케이드 유형(Cascade type)	캐스케이드 루프를 사용하는 경우 슬레이브 또는 마스터(.CT)를 선택합니다.

(1) 래더 기반 PID 명령어를 사용할 때 MAXO = MINO 로 설정하면 PID 명령어는 이 값을 기본값으로 초기화합니다. MAXO = 100.0 및 MINO = 0.0

알람 지정

알람(Alarms) 탭을 선택합니다. 변경 내용은 **확인**(OK) 또는 **적용**(Apply)을 클릭해야 적용됩니다.

해당 필드:	다음을 수행합니다.
PV 상한(PV high)	PV 상한 알람 값(.PVH)을 입력합니다.
PV 하한(PV low)	PV 하한 알람 값(.PVL)을 입력합니다.
PV 불감대(PV deadband)	PV 알람 불감대 값(.PVDB)을 입력합니다.
양의 편차(Positive deviation)	양의 편차 값(.DVP)을 입력합니다.
음의 편차(Negative deviation)	음의 편차 값(.DVN)을 입력합니다.
편차 불감대(Deviation deadband)	편차 알람 불감대 값(.DVDB)을 입력합니다.

스케일링 지정

스케일링(Scaling) 탭을 선택합니다. 변경 내용은 확인(OK) 또는 적용(Apply)을 클릭해야 적용됩니다.

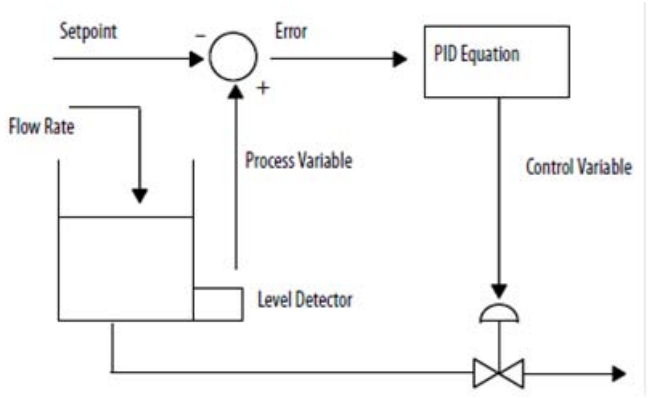
해당 필드:	다음을 수행합니다.
PV 스케일링되지 않은 최대값(PV unscaled maximum)	아날로그 입력 채널에서 PV 값으로 받은 스케일링되지 않은 최대값과 동일한 최대 PV 값(.MAXI)을 입력합니다.
PV 스케일링되지 않은 최소값(PV unscaled minimum)	아날로그 입력 채널에서 PV 값으로 받은 스케일링되지 않은 최소값과 동일한 최소 PV 값(.MINI)을 입력합니다.
PV 엔지니어링 단위 최대값(PV engineering units maximum)	.MAXI 에 해당하는 최대 엔지니어링 단위 (.MAXS)를 입력합니다.
PV 엔지니어링 단위 최소값(PV engineering units minimum)	.MINI 에 해당하는 최소 엔지니어링 단위(.MINS)를 입력합니다.
CV 최대값(CV maximum)	100%에 해당하는 최대 CV 값(.MAXCV)을 입력합니다.

CV 최소값(CV minimum)	0%에 해당하는 최소 CV 값(.MINCV)을 입력합니다.
타이백 최대값(Tieback maximum)	아날로그 입력 채널에서 타이백 값으로 받은 스케일링되지 않은 최대값과 동일한 최대 타이백 값(.MAXTIE)을 입력합니다.
타이백 최소값(Tieback minimum)	아날로그 입력 채널에서 타이백 값으로 받은 스케일링되지 않은 최소값과 동일한 최소 타이백 값(.MINTIE)을 입력합니다.
PID 초기화(PID Initialized)	실행 모드에서 스케일링 상수를 변경하는 경우 이를 해제하여 내부 스케일링 해제 값(.INI)을 다시 초기화할 수 있습니다.

팁: 래더 기반 PID 명령어를 사용할 때 MAXO = MINO 로 설정하면 PID 명령은 이 값을 기본값으로 초기화합니다. MAXO = 100.0 및 MINO = 0.0

PID 명령어 사용

PID 폐쇄 루프 제어는 프로세스 변수를 원하는 설정값으로 유지합니다. 그림에서 유속/유체 수준의 예를 보여줍니다.



위의 예에서 탱크의 레벨은 설정값과 비교됩니다. 레벨이 설정값보다 높으면 PID 수식이 제어 변수를 증가시키고 탱크의 배출 밸브를 열게 됩니다. 따라서 탱크 내의 레벨은 감소됩니다.

PID 명령어에 사용된 PID 수식은 독립 게인 또는 종속 게인 중 하나를 사용하는 옵션이 있는 위치 형식 수식입니다. 독립 게인을 사용할 때 비례, 적분 및 미분 게인은 각각 특정 비례, 적분 또는 미분 항에만 영향을 줍니다. 종속 게인을 사용할 때, 비례 게인은 세 항 모두에 영향을 미치는 컨트롤러 게인으로 대체됩니다. 두 가지 형식의 수식 중 하나를 사용하여 동일한 유형의 제어를 수행할 수 있습니다. 가장 익숙한 수식 유형을 사용할 수 있도록 두 가지 수식 유형이 제공됩니다.

게인 옵션	미분 대상
독립 게인 (ISA 표준)	에러(E)
	프로세스 변수(PV)
종속 게인	에러(E)
	프로세스 변수(PV)

여기에서:

변수	설명
KP	비례 게인(단위 없음) Kp = Kc 단위 없음
Ki	적분 게인(초 -1) Ki(적분 게인) 및 Ti(리셋 시간) 사이에 변환하려면 다음을 사용하십시오. $K_i = \frac{K_C}{60\,T_i}$
Kd	미분 게인(초) Kd(미분 게인) 및 Td(비율 시간) 사이에 변환하려면 다음을 사용하십시오. Kd = Kc (Td) 60
KC	컨트롤러 게인(단위 없음)
Ti	리셋 시간(분/반복)
Td	비율 시간(분)
SP	Setpoint
PV	Process variable
E	에러[(SP-PV) 또는 (PV-SP)]
BIAS	피드포워드 또는 편차
CV	Control variable
dt	루프 업데이트 시간(Loop update time)

PID 수식의 특정 항을 사용하고 싶지 않으면 해당 게인을 0 으로 설정하십시오. 예를 들어, 미분 동작을 원하지 않으면 Kd 또는 Td 를 0 으로 설정하십시오.

추가 참조

무충돌 다시 시작 페이지의 781

미분 평탄화 페이지의 784

불감대 설정 페이지의 789

캐스케이드 루프 페이지의 782

비율 제어 페이지의 782

수동에서 자동으로의 과적분 방지 및 무충돌 전환(PID)

PID 명령어는 .MAXO 와 .MINO 에 설정된 대로 CV 출력이 최대 또는 최소 값에 도달할 때마다 적분항이 누산되는 것을 방지하여 과적분을 자동으로 피합니다. 누산된 적분항은 CV 출력이 최대 제한보다 낮아지거나 최소 제한보다 높아질 때까지 고정된 상태로 유지됩니다. 그 이후에는 적분 누산이 자동으로 다시 시작됩니다.

PID 명령어는 2 가지 수동 제어 모드를 지원합니다.

수동 제어 모드	설명
소프트웨어 수동(.SWM)	이 모드는 설정된 출력 모드라고도 하며, 사용자가 소프트웨어에서 출력 %를 설정할 수 있습니다. 설정된 출력(.SO) 값은 루프의 출력으로 사용됩니다. 설정된 출력 값은 일반적으로 작업자 인터페이스 장치의 작업자 입력에서 가져옵니다.
수동(.MO)	이 모드는 타이백 값을 입력으로 사용하며 내부 변수를 조정하여 출력 시 동일한 값을 생성합니다. PID 명령어에 대한 타이백 입력은 .MINTIE 및 .MAXTIE 의 값에 따라 0 ~ 100%로 스케일링되며 루프의 출력으로 사용됩니다. 타이백 입력은 일반적으로 컨트롤러의 출력을 무시하는 하드웨어 핸드/자동 스테이션의 출력에서 가져옵니다. **중요:** 수동 모드 비트와 소프트웨어 수동 모드 비트를 모두 설정하면 수동 모드가 소프트웨어 수동 모드에 우선합니다.

PID 명령어는 자동으로 소프트웨어 수동 모드에서 자동 모드로 또는 수동 모드에서 자동 모드로 무충돌 전환을 제공합니다. PID 명령어는 CV 출력이 소프트웨어 수동 모드에서 설정된 출력(.SO) 값을 추적하거나 수동 모드에서 타이백 입력을 추적하게 하는 데 필요한 적분항 누산의 값을 역계산합니다. 이러한 방법으로 루프가 자동 모드로 전환되면 CV 출력이 설정된 출력 또는 타이백 값에서 시작되어 출력 값에 범프가 발생하지 않습니다.

또한 PID 명령어는 적분 제어를 사용하지 않을 경우(즉, Ki = 0)에도 자동으로 수동에서 자동으로 무충돌 전환을 제공합니다. 이 경우 이 명령어는 CV 출력이 설정된 출력 또는 타이백 값을 추적하도록 .BIAS 항을 수정합니다. 자동 제어가 다시 시작되면 .BIAS 항이 마지막 값을 유지합니다. PID 데이터 구조에서 .NOBC 비트를 설정하여 .BIAS 항의 역계산을 사용하지 않을 수 있습니다. 단, .NOBC 를 참으로 설정하면 적분 제어가 사용되지 않을 때 PID

명령어가 더 이상 수동에서 자동으로 무충돌 전환을 제공하지 않습니다.

무충돌 다시 시작(PID)

PID 명령어는 1756 아날로그 출력 모듈과 상호 작용하여 컨트롤로가 프로그램에서 실행 모드로 전환하거나 컨트롤러의 전원이 켜지면 무충돌 다시 시작을 지원합니다.

1756 아날로그 출력 모듈은 컨트롤러와의 통신이 끊어지거나 컨트롤러가 프로그램 모드로 설정된 것을 감지할 경우 해당 출력을 사용자가 모듈을 구성할 때 지정한 폴트 조건 값으로 설정합니다. 그런 다음 컨트롤러가 실행 모드로 돌아가거나 아날로그 출력 모듈과의 통신이 다시 연결되면 PID 명령어가 자동으로 PID 명령어의 Inhold bit 및 Inhold Value 파라미터를 사용해 제어 변수 출력을 아날로그 출력과 같아지도록 리셋합니다.

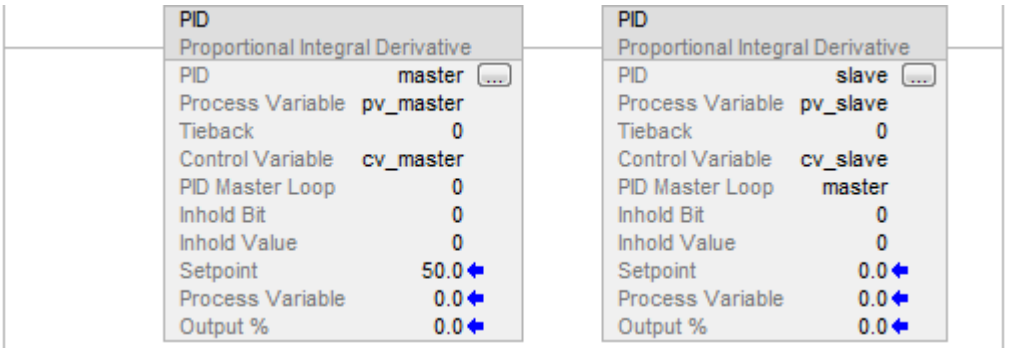
무충돌 다시 시작을 설정하는 명령어

실행할 작업	세부 정보
PID 명령어에서 제어 변수를 수신할 1756 아날로그 출력 모듈의 채널을 구성합니다.	모듈의 특정 채널의 속성 페이지에서 **초기화 동안 보류**(Hold for initialization) 상자를 선택합니다. 그러면 컨트롤러가 실행 모드로 돌아가거나 모듈과의 통신이 다시 연결될 경우 아날로그 출력 모듈이 컨트롤러에서 전송된 값이 출력 채널에 사용된 현재 값과 일치할 때까지(범위의 0.1% 이내) 아날로그 출력을 현재 값으로 유지해야 합니다. 채널의 출력이 .BIAS 항을 사용하여 현재 유지된 출력 값으로 램핑합니다. 이 램핑은 자동 무충돌 전환과 유사합니다.
PID 명령어에 Inhold bit 태그와 Inhold Value 태그를 입력합니다.	1756 아날로그 출력 모듈이 입력 데이터 구조의 각 채널에 대해 2 개의 값을 반환합니다. InHold 상태 비트(예: .Ch2InHold)가 참이면 아날로그 출력 채널이 해당 값을 유지하고 있음을 나타냅니다. 데이터 리드백 값(예: .Ch2Data)은 현재 출력 값을 엔지니어링 단위로 표시합니다. PID 명령어의 InHold bit 파라미터로 InHold 상태 비트의 태그를 입력합니다. Inhold Value 파라미터로 데이터 리드백 값의 태그를 입력합니다. Inhold bit 가 참이면 PID 명렁어가 Inhold Value 를 Control variable 출력으로 이동하고 그 값에서 무충돌 다시 시작을 지원하기 위해 다시 초기화합니다. 아날로그 출력 모듈이 이 값을 컨트롤러에서 다시 수신하면 InHold 상태 비트를 끄므로, PID 명령어가 제어를 정상으로 시작할 수 있습니다.

캐스케이드 루프(PID)

PID 는 마스터 루프의 백분율 단위 출력을 슬레이브 루프의 설정값에 할당하여 두 루프를 캐스케이드식으로 실행합니다. 슬레이브 루프는 .MAXS 및 .MINS 에 대한 슬레이브 루프의 값을 기반으로 마스터 루프의 출력을 자동으로 슬레이브 루프의 설정값에 대해 정확한 엔지니어링 단위로 변환합니다.

릴레이 래더



ST(스트럭처드 텍스트)

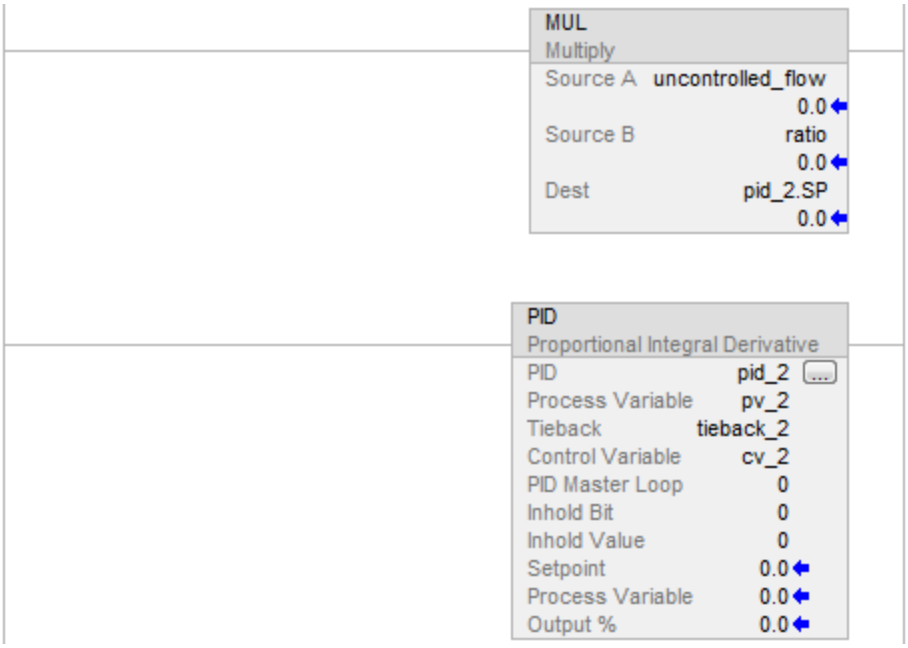
```
PID(master,pv_master,0,cv_master,0,0,0); PID
(slave,pv_slave,0,cv_slave,master,0,0);
```

비율 제어(PID)

이 파라미터를 사용하여 두 값을 한 비율로 유지할 수 있습니다.

- 제어되지 않은 값
- 제어되는 값(PID 명령어가 사용할 결과 설정값)
- 이 두 값 간의 비율

릴레이 래더



팁: PID 를 유효하지 않은 내부 부동 소수점 값으로 고정하는 것을 피하기 위해 다음과 같이 명령어를 호출하기 전에 PV 가 INF 또는 NAN 이 아닌지 확인합니다.

```
XIC (PC_timer.DN)
MOV(Local:0:1.Ch0Data, Local:0:1.Ch0Data)
XIO(S:V)
PID(...)
```

ST(스트럭처드 텍스트)

```
pid_2.sp := uncontrolled_flow * ratio

PID(pid_2,pv_2,tieback_2,cv_2,0,0,0);
```

팁: PID 를 유효하지 않은 내부 부동 소수점 값으로 고정하는 것을 피하기 위해 다음과 같이 명령어를 호출하기 전에 PV 가 INF 또는 NAN 이 아닌지 확인합니다.

```
XIC (PC_timer.DN)
MOV(Local:0:1.Ch0Data, Local:0:1.Ch0Data)
XIO(S:V)
PID(...)
```

이 곱셈을 위해	이 값을 입력
Destination	제어되는 값
Source A	제어되지 않은 값
Source B	Ratio

미분 평탄화(PID)

미분 계산은 미분 평탄화 필터로 강화됩니다. 이 1 차 저역 통과 디지털 필터는 PV 에 있는 노이즈로 인해 발생하는 큰 미분 항 스파이크를 최소화합니다. 이 평탄화는 미분 게인 값이 클 때 더욱 강해집니다. 프로세스에 매우 큰 값의 미분 게인이 필요한 경우, 미분 평탄화를 사용하지 않을 수 있습니다(예: Kd > 10).

미분 평탄화를 사용하지 않으려면

- **구성**(Configuration)탭에서 **미분 평탄화 없음**(No derivative smoothing) 탭을 선택하거나 PID 구조에서 .NDF 비트를 설정합니다.

피드포워드 또는 출력 편차(PID)

.BIAS 값을 PID 명령어의 피드포워드/편차 값으로 전달하여 시스템에서 외란을 피드포워드합니다.

피드포워드 값은 외란이 프로세스 변수를 변경할 수 있게 되기 전에 PID 명령어에 전달된 외란을 나타냅니다. 피드포워드는 흔히 프로세스를 제어하기 위해 전송 래그와 함께 사용됩니다. 예를 들어 '온수에 따른 냉수'를 나타내는 피드포워드 값은 프로세스 변수가 혼합의 결과로 변경되기를 기다리는 것보다 빨리 출력 값을 증가할 수 있습니다.

편차 값은 일반적으로 적분 제어를 사용하지 않을 때 사용됩니다. 이 경우 PV 를 설정값 부근으로 유지하는 데 필요한 범위에서 출력을 유지하도록 편차 값을 조정할 수 있습니다.

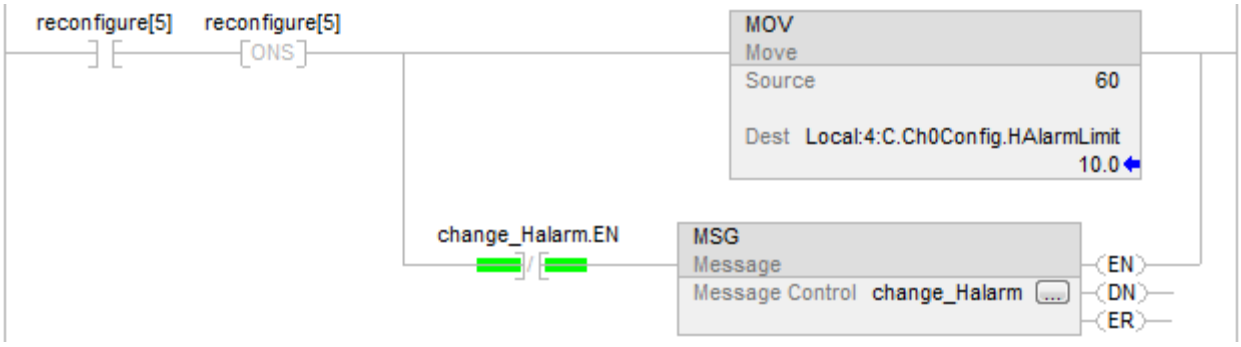
PID 명령어 타이밍

PID 명령어 및 프로세스 변수의 샘플링은 주기적으로 업데이트해야 합니다. 이 업데이트 시간은 제어 중인 물리적 프로세스와 관계가 있습니다. 온도 루프와 같이 매우 느린 루프의 경우 일반적으로 초당 한 번 또는 그보다 더 긴 업데이트 시간이라도 좋은 제어를 얻기에 충분합니다. 압력 또는 흐름 루프와 같이 좀 더 빠른 루프는 매 250 ms 마다 한 번과 같은 업데이트 시간이 필요할 수 있습니다. 언와인드 스풀에 대한 장력 제어와 같이 드문 경우에 매 10 ms 또는 그보다 짧은 시간마다 한 번의 루프 업데이트가 필요합니다.

PID 명령어는 해당 계산에 시간 기준을 사용하기 때문에 이 명령어 실행을 프로세스 변수(PV)의 샘플링과 동기화해야 합니다.

PID 명령어를 실행하는 가장 쉬운 방법은 PID 명령어를 주기 태스크에 넣는 것입니다. 루프 업데이트 시간(.UPD)을 주기 태스크 속도와 같게 설정하고 PID 명령어를 주기적 태스크 스캔마다 실행하도록 합니다.

릴레이 래더



팁: PID 를 유효하지 않은 내부 부동 소수점 값으로 고정하는 것을 피하기 위해 다음과 같이 명령어를 호출하기 전에 PV 가 INF 또는 NAN 이 아닌지 확인합니다.

```
XIC (PC_timer.DN)
MOV(Local:0:1.Ch0Data, Local:0:1.Ch0Data)
XIO(S:V)
PID(...)
```

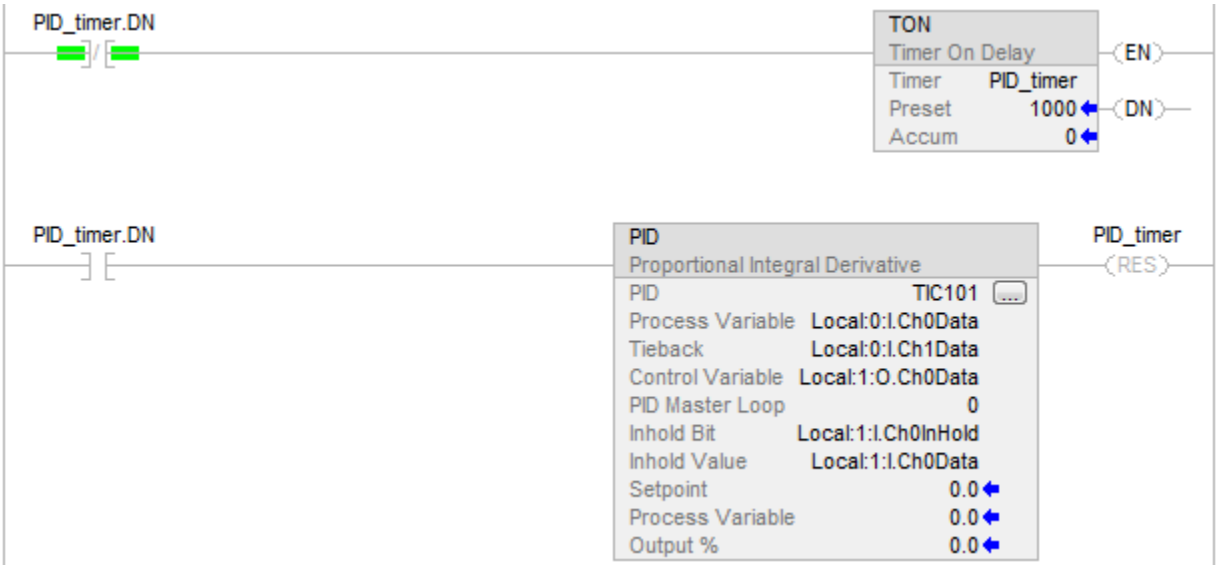
ST(스트럭처드 텍스트)

```
PID(TIC101,Local:0:I.Ch0Data,Local:0:I.Ch1Data,
Local:1:O.Ch4Data,0,Local:1:I.Ch4InHold, Local:1:I.Ch4Data);
```

주기 태스크를 사용하는 경우 프로세스 변수에 사용하는 아날로그 입력을 주기적 태스크의 속도보다 훨씬 더 빠른 속도로 프로세서에 대해 업데이트해야 합니다. 이상적인 경우 프로세서 변수를 주기 태스크 속도보다 최소 5 ~ 10 배 더 빠르게 프로세서에 보내야 합니다. 이렇게 하면 프로세스 변수의 실제 샘플과 PID 루프의 실행 간에 시간 차이는 최소화합니다. 예를 들어 PID 루프가 250 ms 주기 태스크 내에 있는 경우 250 ms(.UPD = .25)의 루프 업데이트 시간을 사용하고 아날로그 입력 모듈을 최소 약 25 ~ 50 ms 마다 데이터를 생성하도록 구성합니다.

정확성이 다소 떨어지는 다른 PID 명령어 실행 방법은 명령어를 연속 태스크에 배치하고 타이머 완료 비트를 사용하여 PID 명령어의 실행을 트리거하는 것입니다.

릴레이 래더



팁: PID 를 유효하지 않은 내부 부동 소수점 값으로 고정하는 것을 피하기 위해 다음과 같이 명령어를 호출하기 전에 PV 가 INF 또는 NAN 이 아닌지 확인합니다.

```
XIC (PC_timer.DN)
MOV(Local:0:1.Ch0Data, Local:0:1.Ch0Data)
XIO(S:V)
PID(...)
```

ST(스트럭처드 텍스트)

```
PID_timer.pre := 1000

TONR(PID_timer);

IF PID_timer.DN THEN PID(TIC101,Local:0:I.Ch0Data,Local:0:I.Ch1Data,

Local:1:O.Ch0Data,0,Local:1:I.Ch0InHold,

Local:1:I.Ch0Data);

END_IF;
```

팁: PID 를 유효하지 않은 내부 부동 소수점 값으로 고정하는 것을 피하기 위해 다음과 같이 명령어를 호출하기 전에 PV 가 INF 또는 NAN 이 아닌지 확인합니다.

```
XIC (PC_timer.DN)
MOV(Local:0:1.Ch0Data, Local:0:1.Ch0Data)
XIO(S:V)
PID(...)
```

이 방법에서 PID 명령어의 루프 업데이트 시간은 타이머 미리 설정 값과 같도록 설정해야 합니다. 주기 태스크를 사용하는 경우와 같이 아날로그 입력 모듈을 루프 업데이트 시간보다 훨씬 더 빠른 속도로 프로세스 변수를 생성하도록 설정해야 합니다. PID 실행의 타이머 방법은 연속 태스크에 대한 최악의 경우의 실행 시간보다 최소 몇 배 더 긴 루프 업데이트 시간을 가진 루프에만 사용해야 합니다.

PID 명령어를 실행하는 가장 정확한 방법은 1756 아날로그 입력 모듈의 실시간 샘플링(RTS) 기능을 사용하는 것입니다. 아날로그 입력 모듈은 모듈을 설정할 때 구성한 실시간 샘플링 속도로 해당 입력을 샘플링합니다. 모듈의 실시간 샘플링 기간이 만료하면 입력을 업데이트하고 모듈이 생성한 롤링 타임스탬프(아날로그 입력 데이터 구조의 .RollingTimestamp 구성원에 의해 표시)를 업데이트합니다.

타임스탬프 범위는 0 ~ 32,767 ms 입니다. 타임스탬프를 모니터합니다. 변경되면 새로운 프로세스 변수 샘플이 수신됩니다. 타임스탬프가 변경될 때마다 PID 명령어를 한 번 실행하십시오. 프로세스 변수 샘플은 아날로그 입력 Module 에 의해 구동되므로 입력 샘플 시간은 매우 정확하며 PID 명령어에 사용되는 루프 업데이트 시간은 아날로그 입력 모듈의 RTS 시간과 동일하게 설정해야 합니다.

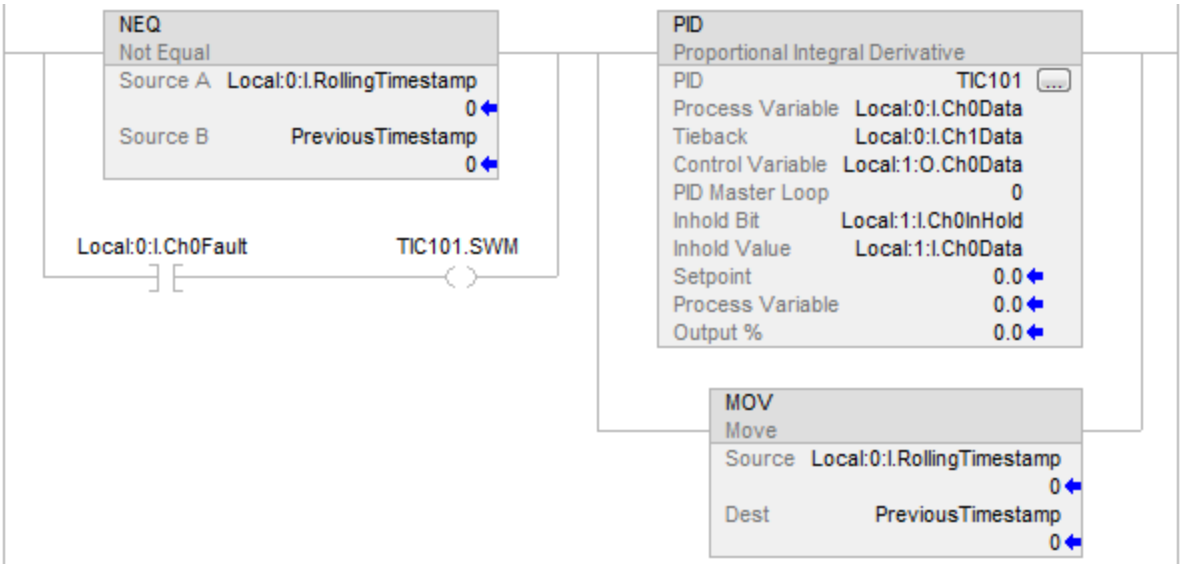
프로세스 변수 샘플을 누락하지 않으려면 RTS 시간보다 빠른 속도로 로직을 실행하십시오. 예를 들어, RTS 시간이 250 ms 일 경우 PID 로직을 100 ms 마다 실행되는 주기 태스크에 배치하여

샘플을 누락하지 않을 수 있습니다. 로직이 250 ms 마다 두 번 이상 자주 업데이트되도록 하는 한 PID 로직을 연속 태스크에 배치할 수도 있습니다.

RTS 실행 방법의 예는 아래와 같습니다. PID 명령어의 실행은 새로운 아날로그 입력 데이터 수신에 따라 다릅니다. 아날로그 입력 모듈이 실패하거나 제거되면 컨트롤러는 롤링 타임스탬프 수신을 중지하고 PID 루프 실행이 중지됩니다. PV 아날로그 입력의 상태 비트를 모니터링하고 상태가 좋지 않으면 루프를

소프트웨어 수동 모드로 강제 실행하고 매 스캔마다 루프를 실행해야 합니다. 이를 통해 조작자는 PID 루프의 출력을 수동으로 변경할 수 있습니다.

릴레이 래더

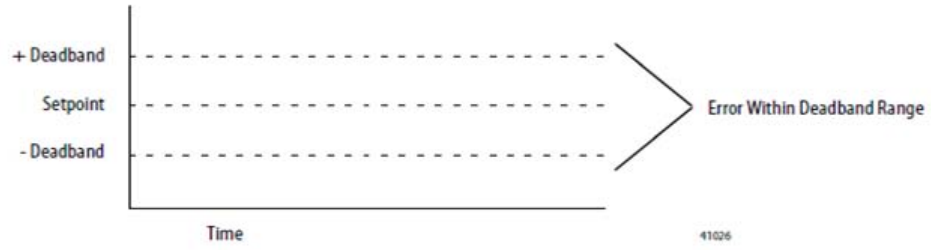


ST(스트럭처드 텍스트)

```
IF (Local:0:I.Ch0Fault) THEN TIC101.SWM [:=] 1;

ELSE TIC101.SWM := 0; END_IF;

IF (Local:0:I.RollingTimestamp<>PreviousTimestamp) OR
(Local:0:I.Ch0Fault) THEN

PreviousTimestamp := Local:0:I.RollingTimestamp;
PID(TIC101,Local:0:I.Ch0Data,Local:0:I.Ch1Data,

Local:1:O.Ch0Data,0,Local:1:I.Ch0InHold,

Local:1:I.Ch0Data);

END_IF;
```

불감대 설정(PID)

조절 가능한 불감대를 통해 설정값 위와 아래의 에러 범위를 선택하여 에러가 이 범위 내에 있으면 출력이 변경되지 않도록 할 수 있습니다. 이 불감대로 출력 변경 없이 프로세서 변수가 설정값과 얼마나 근접하게 일치하는지 제어할 수 있습니다. 불감대는 최종 제어 장치의 손상을 최소화하는 데도 도움이 됩니다.



0 점 교차는 프로세스 변수가 설정값을 통과할 때까지 프로세스 변수가 불감대로 교차함에 따라 명령어가 계산 목적으로 에러를 사용할 수 있게 하는 불감대 제어입니다. 프로세스 변수가 설정값을 넘으면(에러는 0 을 교차하고 부호를 변경함) 프로세스 변수가 불감대에 남아있는 한 출력은 변경되지 않습니다.

불감대는 사용자가 지정한 값만큼 설정값 위아래로 확장됩니다. 불감대를 금지하려면 0 을 입력합니다. 불감대의 스케일링된 단위는 설정값과 동일합니다. **구성**(Configuration) 탭에서 **불감대에 0 점 교차 없음**(No zero crossing for deadband)을 선택하여 0 점 교차 기능 없이 불감대를 사용하여 PID 구조에서 .NOZC 비트를 설정합니다.

불감대를 사용하는 경우, Control variable 은 REAL 이어야 하며 에러가 데드밴드 내에 있을 경우 강제로 0 이 됩니다.

불감대를 금지하려면:

- 0 을 입력합니다.

불감대의 스케일링된 단위는 설정값과 동일합니다.

0 점 교차 기능 없이 불감대를 사용하려면:

- **구성**(Configuration) 탭에서 **불감대에 0 점 교차 없음**(No zero crossing for deadband)을 선택하거나 PID 구조에서 .NOZC 비트를 설정합니다.

불감대를 사용하는 경우, Control variable 은 REAL 이어야 하며 에러가 불감대 내에 있을 경우 강제로 0 이 됩니다.

출력 제한 사용(PID)

제어 출력에 대해 출력 제한(출력 비율)을 설정합니다. 명령어가 출력이 한계에 도달했음을 감지하면 알람 비트를 설정하고 출력이 하한값 또는 상한값을 초과하는 것을 방지합니다.

삼각법 명령어

삼각함수 명령어는 삼각함수 연산을 사용하여 산술 연산을 계산합니다.

사용 가능한 명령어

래더 다이어그램, 평션 블록 및 ST(스트럭처드 텍스트)

SIN	ATN, ATAN	COS	TAN	ASN, ASIN	ACS/ASOS

실행할 작업:	사용할 명령어:
값의 사인을 구합니다.	SIN
값의 코사인을 구합니다.	COS
값의 탄젠트를 구합니다.	TAN
값의 아크 사인을 구합니다.	ASN
값의 아크 코사인을 구합니다.	ACS
값의 아크 탄젠트를 구합니다.	ATN

데이터 유형을 혼합 사용할 수 있지만 정확도가 떨어지고 반올림 에러가 발생할 수 있으며 명령어를 실행하는 데 시간이 더 오래 걸립니다. 결과가 잘렸는지 확인하려면 S:V 비트를 확인하십시오.

볼드체 데이터 유형은 최적의 데이터 유형을 나타냅니다. 명령어의 모든 피연산자가 동일한 최적의 데이터 유형(일반적으로 DINT 또는 REAL)을 사용하는 경우 명령어는 가장 적은 메모리를 사용하면서 보다 빠르게 실행됩니다.

삼각함수 명령어는 렁-입력-조건이 참인 경우 명령어가 스캔될 때마다 한 번 실행됩니다. 명령어를 한 번만 계산하려면 ONS 명령어를 사용하여 삼각함수 명령어를 트리거하십시오.

추가 참조

타이머 및 카운터 명령어 페이지의 111

특수 명령어 페이지의 745

시퀀서 명령어 페이지의 663

프로그램 제어 명령어 페이지의 680

이동/로직 명령어 페이지의 475

삼각법 명령어

삼각함수 명령어는 삼각함수 연산을 사용하여 산술 연산을 계산합니다.

사용 가능한 명령어

래더 다이어그램, 평션 블록 및 ST(스트럭처드 텍스트)

SIN	ATN, ATAN	COS	TAN	ASN, ASIN	ACS/ASOS

실행할 작업:	사용할 명령어:
값의 사인을 구합니다.	SIN
값의 코사인을 구합니다.	COS
값의 탄젠트를 구합니다.	TAN
값의 아크 사인을 구합니다.	ASN
값의 아크 코사인을 구합니다.	ACS
값의 아크 탄젠트를 구합니다.	ATN

데이터 유형을 혼합 사용할 수 있지만 정확도가 떨어지고 반올림 에러가 발생할 수 있으며 명령어를 실행하는 데 시간이 더 오래 걸립니다. 결과가 잘렸는지 확인하려면 S:V 비트를 확인하십시오.

볼드체 데이터 유형은 최적의 데이터 유형을 나타냅니다. 명령어의 모든 피연산자가 동일한 최적의 데이터 유형(일반적으로 DINT 또는 REAL)을 사용하는 경우 명령어는 가장 적은 메모리를 사용하면서 보다 빠르게 실행됩니다.

삼각함수 명령어는 렁-입력-조건이 참인 경우 명령어가 스캔될 때마다 한 번 실행됩니다. 명령어를 한 번만 계산하려면 ONS 명령어를 사용하여 삼각함수 명령어를 트리거하십시오.

추가 참조

타이머 및 카운터 명령어 페이지의 111

특수 명령어 페이지의 745

시퀀서 명령어 페이지의 663

프로그램 제어 명령어 페이지의 680

이동/로직 명령어 페이지의 475

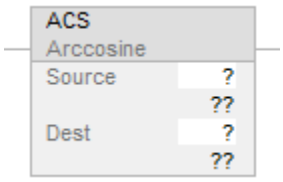
아크 코사인(ACS, ACOS)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

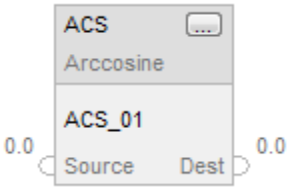
ACS 명령어는 Source 값의 아크 코사인을 구하고 결과를 Destination 에 저장합니다(라디안 단위).

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

dest := ACOS(source);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 코사인을 구함
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 코사인을 구함

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

ACOS 를 함수로 사용합니다. 이 함수는 소스의 아크 코사인을 계산하고 REAL 결과를 반환합니다.

평션 블록

피연산자	유형	형식	설명
ACS tag	FBD_MATH_ADVANCED	구조	ACS 구조

FBD_MATH_ADVANCED 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
Source	REAL	연산 명령어에 대한 입력.

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
Dest	REAL	연산 명령어의 결과.

설명

ACS 명령어는 소스 값의 아크 코사인을 구하고 REAL 결과를 반환하고 Destination 에 저장합니다(라디안 단위). Source 는 -1 이상 1 이하여야 합니다. Destination 의 결과 값은 0 이상 pi 이하입니다. Source 가 -1 보다 작거나 1 보다 크면 Destination 이 NAN 으로 설정됩니다.

ACS 를 래더 식의 연산자로 사용할 수 있으며, ACOS 를 ST(스트럭쳐드 텍스트) 문의 연산자로 사용할 수 있습니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
ControlLogix 5580	조건부, 연산 상태 플래그 참조하십시오.
CompactLogix 5370, ControlLogix 5570	예

메이저/마이너 폴트

대상이 NAN 으로 설정된 경우 해당 조건부 마이너 폴트 포함 오버플로가 생성됩니다.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	컨트롤러는 Source 의 아크 코사인을 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

평션 블록

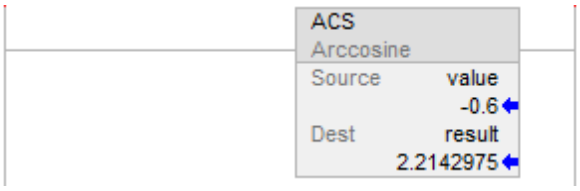
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

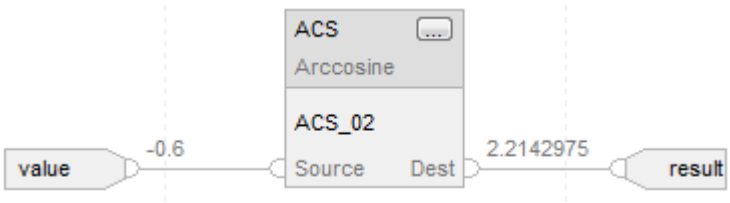
조건/상태	취해진 조치
사전 스캔	N/A.
정상 실행	컨트롤러는 Source 의 아크 코사인을 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

예

래더 다이어그램



평션 블록



ST(스트럭쳐드 텍스트)

result := ACOS(value);

추가 참조

삼각법 명령어 페이지의 792

공통 속성 페이지의 963

연산 상태 플래그 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

아크 사인(ASN, ASIN)

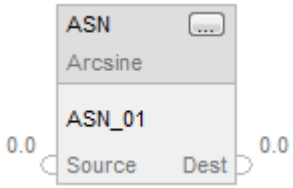
이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

ASN 명령어는 Source 값의 아크 사인을 구하고 결과를 Destination 에 저장합니다(라디안 단위).

사용 가능한 언어

래더 다이어그램

평션 블록



ST(스트럭처드 텍스트)

dest :=ASIN(source);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT **REAL**	즉시 태그	이 값의 아크 사인을 구함
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 아크 사인을 구함

ST(스트럭쳐드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

ASIN 을 함수로 사용합니다. 이 함수는 소스의 아크사인을 계산하고 REAL 결과를 반환합니다.

평션 블록

피연산자	유형	형식	설명
ASN tag	FBD_MATH_ADVANCED	구조	ASN 구조

FBD_MATH_ADVANCED 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
Source	REAL	연산 명령어에 대한 입력. 유효값 = 모든 부동 소수점 수

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
Dest	REAL	명령어의 결과.

설명

ASN 명령어는 Source 값의 아크 사인을 구하고 REAL 결과를 반환하고 Destination 에 저장합니다(라디안 단위). Source 는 -1 이상 1 이하여야 합니다. Destination 의 결과 값은 -pi/2 이상 pi/2 이하입니다. Source 가 -1 보다 작거나 1 보다 크면 Destination 이 NAN 으로 설정됩니다.

ASN 을 래더 식의 연산자로 사용할 수 있으며, ASIN 을 ST(스트럭쳐드 텍스트) 문의 연산자로 사용할 수 있습니다.

이 명령어는 더 나은 결과를 위해 레거시 컨트롤러에 대해 더 나은 정확도를 제공합니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, *연산 상태 플래그* 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

대상이 NAN 으로 설정된 경우 해당 조건부 마이너 폴트 포함 오버플로가 생성됩니다.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

펑션 블록

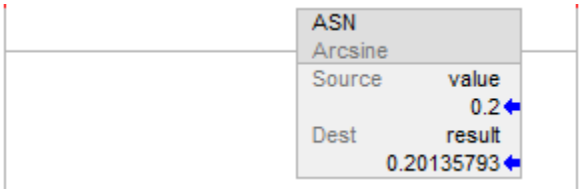
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

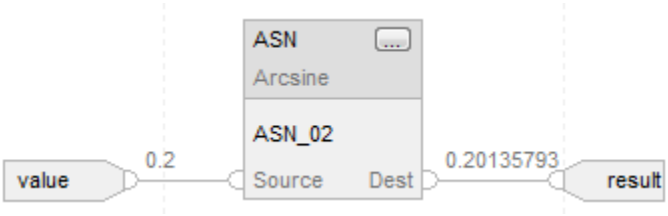
조건/상태	취해진 조치
사전 스캔	N/A.
정상 실행	컨트롤러는 Source 의 아크사인을 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

예

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

```
result := ASIN(value);
```

추가 참조

삼각법 명령어 페이지의 792

공통 속성 페이지의 963

연산 상태 플래그 페이지의 963

ST(스트럭처드 텍스트) 구문 페이지의 997

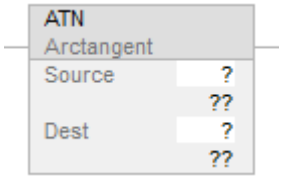
아크 탄젠트(ATN, ATAN)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

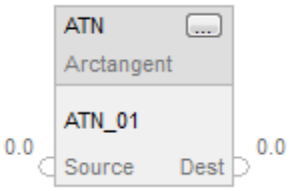
ATN 명령어는 Source 값의 아크 탄젠트를 계산하고 결과를 Destination 에 저장합니다(라디안 단위).

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

```
dest := ATAN(source);
```

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 아크 탄젠트를 구함
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 아크 탄젠트를 구함

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

ATAN 을 함수로 사용합니다. 이 함수는 소스의 아크 탄젠트를 계산하고 REAL 결과를 반환합니다.

펑션 블록

피연산자	유형	형식	설명
ATN tag	FBD_MATH_ADVANCED	구조	ATN 구조

FBD_MATH_ADVANCED 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
Source	REAL	연산 명령어에 대한 입력. 유효값 = 모든 부동 소수점 수

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
Dest	REAL	명령어의 결과.

설명

ATN 명령어는 Source 값의 아크 탄젠트를 계산하고 결과를 Destination 에 저장합니다(라디안 단위). Destination 의 결과 값은 -pi/2 이상 pi/2 이하입니다.

ATN 을 래더 식의 연산자로 사용할 수 있으며, ATAN 을 ST(스트럭처드 텍스트) 문의 연산자로 사용할 수 있습니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, 연산 상태 플래그 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	컨트롤러는 Source 의 아크 탄젠트를 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

평션 블록

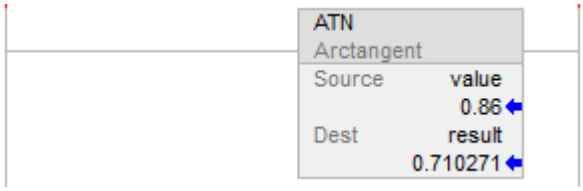
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

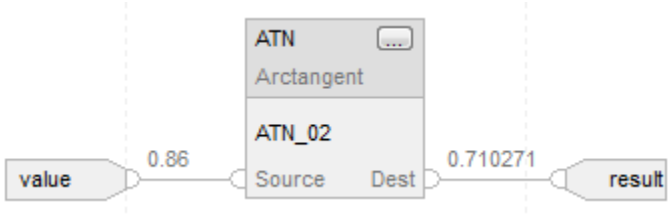
조건/상태	취해진 조치
사전 스캔	N/A
정상 실행	컨트롤러는 Source 의 아크 탄젠트를 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

예

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

```
result := ATAN(value);.
```

추가 참조

삼각법 명령어 페이지의 792

공통 속성 페이지의 963

데이터 변환 페이지의 967

연산 상태 플래그 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

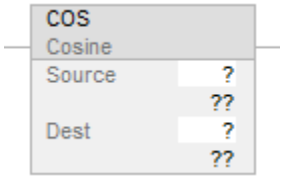
코사인(COS)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

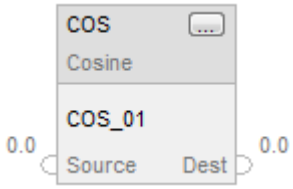
COS 명령어는 Source 값의 코사인(라디안 단위)을 구하고 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록



ST(스트럭쳐드 텍스트)

```
dest := COS(source);
```

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 코사인을 구함
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 코사인을 구함

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

평션 블록

피연산자	유형	형식	설명
COS tag	FBD_MATH_ADVANCED	구조	COS 구조

FBD_MATH_ADVANCED 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
Source	REAL	연산 명령어에 대한 입력.

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
Dest	REAL	연산 명령어의 결과.

설명

COS 명령어는 Source 값의 코사인(라디안 단위)을 계산하고 결과를 Destination 에 저장합니다.

이 명령어는 Source 의 코사인을 계산하고 REAL 결과를 반환합니다. 결과 값은 언제나 -1 이상 1 이하입니다.

COS 를 래더 식의 연산자 및 ST(스트럭처드 텍스트) 문의 연산자로 사용할 수 있습니다.

이 명령어는 더 나은 결과를 위해 레거시 컨트롤러에 대해 더 나은 정확도를 제공합니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, 연산 상태 플래그 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

없음. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A.
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	컨트롤러는 Source 의 코사인을 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

평션 블록

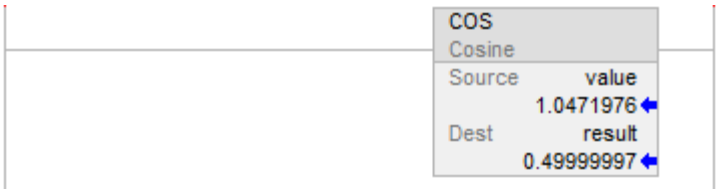
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

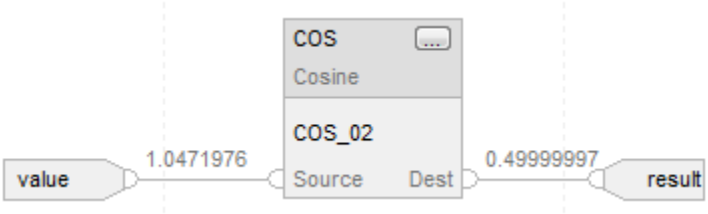
조건/상태	취해진 조치
사전 스캔	N/A.
정상 실행	컨트롤러는 Source 의 코사인을 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

예

래더 다이어그램



평션 블록



ST(스트럭쳐드 텍스트)

```
result := COS(value);
```

추가 참조

삼각법 명령어 페이지의 792

라디안(RAD) 페이지의 850

공통 속성 페이지의 963

연산 상태 플래그 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

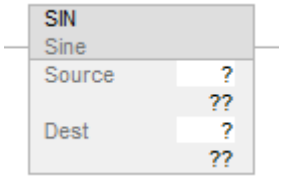
사인(SIN)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

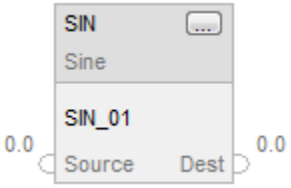
SIN 명령어는 Source 값의 사인(라디안 단위)을 구하고 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

```
dest := SIN(source);
```

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
소스	SINT INT DINT REAL	즉시 태그	이 값의 사인을 구함
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
소스	SINT INT DINT REAL	즉시 태그	이 값의 사인을 구함

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

펑션 블록

피연산자	유형	형식	설명
SIN tag	FBD_MATH_ADVANCED	Structure	SIN 구조

FBD_MATH_ADVANCED 구조

입력 파라미터	데이터 유형	설명
활성화 입력	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
소스	REAL	연산 명령어에 대한 입력.

출력 파라미터	데이터 유형	설명
활성화 출력(EnableOut)	BOOL	명령어의 활성화 여부를 나타냅니다.
Dest	REAL	연산 명령어의 결과.

연산자 특성

SIN 연산자는 다양한 식에 사용할 수 있습니다. 마찬가지로 SIN 함수는 ST(스트럭처드 텍스트) 문에서 호출됩니다. SIN 의 두 가지 적용 모두 Source 의 사인을 포함한 REAL 결과를 반환합니다. 컨텍스트에 따라 해당하는 경우 이 값이 형 변환될 수 있습니다.

설명

SIN 명령어는 Source 값의 사인(라디안 단위)을 구하고 결과를 Destination 에 저장합니다.

이 명령어는 Source 의 사인을 계산하고 REAL 결과를 반환합니다. 결과 값은 언제나 -1 이상 1 이하입니다.

SIN 을 래더 식의 연산자 및 ST(스트럭쳐드 텍스트) 문의 함수로 사용할 수 있습니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, *연산 상태 플래그* 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A.
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	컨트롤러는 Source 의 사인을 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

펑션 블록

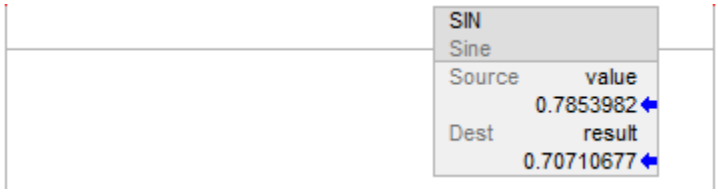
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

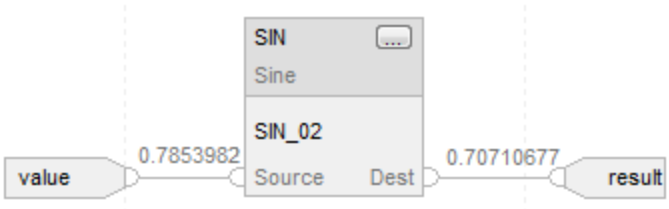
조건/상태	취해진 조치
사전 스캔	N/A.
정상 실행	컨트롤러는 Source 의 사인을 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

예

래더 다이어그램



펑션 블록



ST(스트럭처드 텍스트)

```
result := SIN(value);
```

추가 참조

삼각법 명령어 페이지의 792

공통 특성 페이지의 963

연산 상태 플래그 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

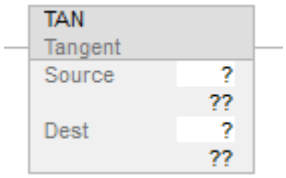
탄젠트(TAN)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

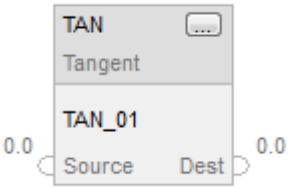
TAN 명령어는 Source 값의 탄젠트(라디안 단위)를 구하고 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭쳐드 텍스트)

```
dest := TAN(source);
```

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 코사인을 구함
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 탄젠트를 구함

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

펑션 블록

피연산자	유형	형식	설명
TAN tag	FBD_MATH_ADVANCED	구조	TAN 구조

FBD_MATH_ADVANCED 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
Source	REAL	연산 명령어에 대한 입력.

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어의 활성화 여부를 나타냅니다.
Dest	REAL	연산 명령어의 결과.

설명

TAN 명령어는 Source 값의 탄젠트(라디안 단위)를 구하고 결과를 Destination 에 저장합니다.

이 명령어는 Source 의 탄젠트를 계산하고 REAL 결과를 반환합니다.

TAN 을 래더 식의 연산자 및 ST(스트럭처드 텍스트) 문의 연산자로 사용할 수 있습니다.

이 명령어는 더 나은 결과를 위해 레거시 컨트롤러에 대해 더 나은 정확도를 제공합니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부임. 연산 상태 플래그 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	컨트롤러는 Source 의 탄젠트를 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

평션 블록

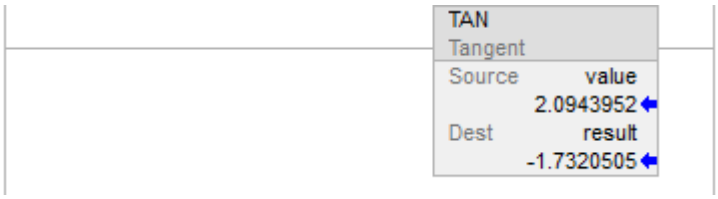
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건/상태	취해진 조치
사전 스캔	N/A.
정상 실행	컨트롤러는 Source 의 탄젠트를 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

예

래더 다이어그램



평션 블록

ST(스트럭쳐드 텍스트)

result := TAN(value);

추가 참조

삼각법 명령어 페이지의 792

공통 속성 페이지의 963

연산 상태 플래그 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

고급 연산

고급 연산 명령어

고급 수학 명령어에는 다음 명령어가 포함됩니다.

래더 다이어그램 및 평션 블록

LN	LOG	XPY

ST(스트럭처드 텍스트)

LN	LOG	XPY

실행할 작업:	사용할 명령어:
값의 자연 로그를 구할 경우	LN
값의 로그 밑수 10 을 구할 경우	LOG
값을 다른 값의 제곱으로 올릴 경우	XPY

데이터 유형을 혼합하면 정확도 및 반올림 에러가 발생할 수 있으며 명령어 실행이 오래 걸릴 수 있습니다. 결과가 잘렸는지 확인하려면 S:V 비트를 확인하십시오.

볼드체 데이터 유형은 최적의 데이터 유형을 나타냅니다. 명령어의 모든 피연산자가 동일한 최적의 데이터 유형(일반적으로 DINT 또는 REAL)을 사용하는 경우 명령어는 가장 적은 메모리를 사용하면서 보다 빠르게 실행됩니다.

고급 수학 명령어는 렁-입력-조건이 참인 경우 명령어가 스캔될 때마다 한 번 실행됩니다. 명령어를 한 번만 실행하려면 ONS 명령어를 사용하여 수학 명령어를 트리거하십시오.

추가 참조

배열(파일)/기타 명령어 페이지의 545

ASCII 변환 명령어 페이지의 927

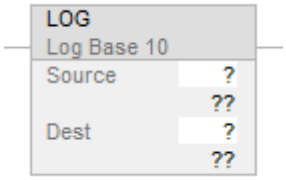
로그 밑수 10(LOG)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

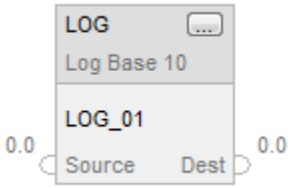
LOG 명령어는 Source 의 로그 밑수 10 을 구하고 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록



ST(스트럭처드 텍스트)

```
dest := LOG(source);
```

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 로그를 구함
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

LOG 를 함수로 사용합니다. 이 함수는 소스의 로그를 계산하고 결과를 대상에 저장합니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

펑션 블록

피연산자	유형	형식	설명
LOG tag	FBD_MATH_ADVANCED	구조	LOG 구조

FBD_MATH_ADVANCED 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
Source	REAL	연산 명령어에 대한 입력.

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	출력 활성화.
Dest	REAL	연산 명령어의 결과. 이 출력에 대해 연산 상태 플래그가 설정됩니다.

설명

LOG 명령어는 Source 의 로그 밑수 10 을 구하고 결과를 Destination 에 저장합니다. Source 는 0 보다 커야 하며 그렇지 않으면 마이너 폴트가 발생합니다.

Source	Destination
숫자가 아님 음수 음의 무한대,	숫자가 아님, 오버플로 마이너 폴트 발생
0 음수 양수	음의 무한대, 오버플로 마이너 폴트 발생
양수	정상 결과
양의 무한대	양의 무한대, 오버플로 마이너 폴트 발생

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, 연산 상태 플래그 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A.
렁-입력-조건이 거짓임	N/A.
렁-입력-조건이 참임	컨트롤러는 Source 의 자연 로그를 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A.

평션 블록

조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

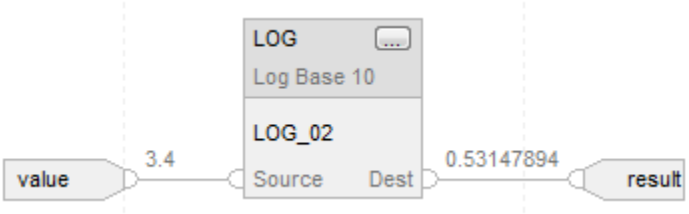
조건/상태	취해진 조치
사전 스캔	N/A.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	N/A.

예

값의 로그를 계산하고 결과를 result 에 놓습니다.

래더 다이어그램

평션 블록



ST(스트럭쳐드 텍스트)

result := LOG(value);

추가 참조

공통 속성 페이지의 963

고급 연산 명령어 페이지의 821

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

ST(스트럭쳐드 텍스트) 구문 페이지의 997

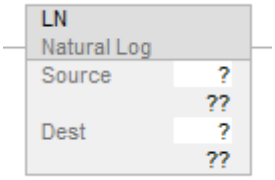
자연 로그(LN)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

LN 명령어는 Source 의 자연 로그를 구하고 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

ST(스트럭처드 텍스트)

```
dest := LN(source);
```

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	이 값의 자연 로그를 구함
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

LN 을 함수로 사용합니다. 이 함수는 소스의 자연 로그를 계산하고 결과를 대상에 저장합니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

평션 블록

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	출력 활성화.
Dest	REAL	연산 명령어의 결과. 이 출력에 대해 연산 상태 플래그가 설정됩니다.

FBD_MATH_ADVANCED 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
Source	REAL	연산 명령어에 대한 입력.

설명

LN 명령어는 Source 의 자연 로그를 구하고 결과를 Destination 에 저장합니다. Source 는 0 보다 커야 하며 그렇지 않으면 마이너 폴트가 발생합니다.

다음 표는 부동 소수점 소스 값에 대한 특별한 경우를 보여 줍니다.

Source	Destination
숫자가 아님 음수 음의 무한대,	숫자가 아님, 오버플로 마이너 폴트 발생
0 음수 양수	음의 무한대, 오버플로 마이너 폴트 발생
양의 무한대	양의 무한대, 오버플로 마이너 폴트 발생

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, *연산 상태 플래그* 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	컨트롤러는 Source 의 자연 로그를 계산하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

펑션 블록

조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

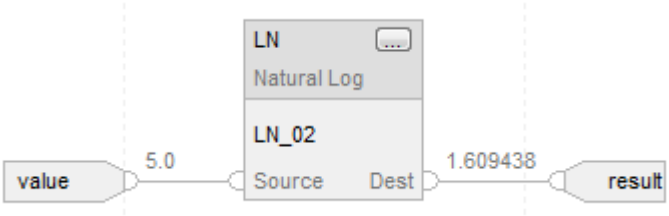
조건/상태	취해진 조치
사전 스캔	N/A.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	N/A.

예

값의 자연 로그를 계산하고 결과를 result 에 놓습니다.

래더 다이어그램

평션 블록



ST(스트럭처드 텍스트)

```
result := LN(value);
```

추가 참조

고급 연산 명령어 페이지의 821

공통 속성 페이지의 963

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

ST(스트럭쳐드 텍스트) 구문 페이지의 997

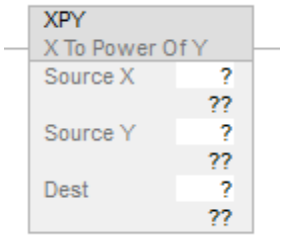
X 의 Y 제곱(XPY)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

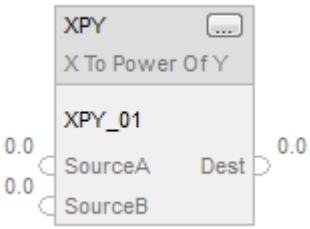
XPY 명령어는 Source A(X)의 Source B(Y)제곱 구하여 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록



ST(스트럭쳐드 텍스트)

```
dest := sourceX ** sourceY;
```

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source X	SINT INT DINT REAL	immediate 태그	밑수
Source Y	SINT INT DINT REAL	immediate 태그	지수
Dest	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

인접한 두 곱하기 기호 "**"를 식 내의 연산자로 사용합니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

펑션 블록

피연산자	유형	형식	설명
XPY tag	FBD_MATH	Structure	XPY 구조

FBD_MATH 구조

입력 파라미터	데이터 유형	설명
활성화 입력	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
SourceA	REAL	밑수 값.
SourceB	REAL	지수.

출력 파라미터	데이터 유형	설명
활성화 출력(EnableOut)	BOOL	출력 활성화.
Dest	REAL	연산 명령어의 결과. 이 출력에 대해 연산 상태 플래그가 설정됩니다.

설명

XPY 명령어는 Source A(X)의 Source B(Y)제곱 구하여 결과를 Destination 에 저장합니다.

Source A(X)가 음수이면 Source B(Y)는 분수가 아닌 값이거나 마이너 폴트가 발생합니다.

CompactLogix 5370 및 ControlLogix 5570 컨트롤러의 경우, 밑수가 음수이고 지수가 실수이면 밑수의 절대값을 사용합니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, *연산 상태 플래그* 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

컨트롤러	메이저 폴트는 다음과 같은 경우에 발생합니다.	폴트 유형	폴트 코드
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	N/A	N/A	N/A
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	Source X 가 음수이고 Source Y 가 정수 값이 아님	4	4

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A.
렁-입력-조건이 거짓임.	N/A.
렁-입력-조건이 참임.	컨트롤러는 Source X 의 Source Y 제곱을 구하여 결과를 Destination 에 놓습니다.
사후 스캔	N/A.

평션 블록

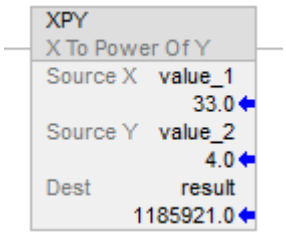
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

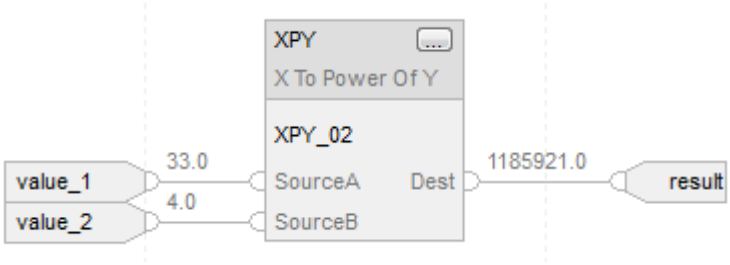
조건/상태	취해진 조치
사전 스캔	N/A.
정상 실행	렁-입력-조건이 참을 참조하십시오.
사후 스캔	N/A.

예

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

```
result := (value_1 ** value_2);
```

추가 참조

ST(스트럭쳐드 텍스트) 구문 페이지의 997

고급 연산 명령어 페이지의 821

연산 상태 플래그 페이지의 963

공통 특성 페이지의 963

연산 변환 명령어

연산 변환 명령어

수학 변환 명령어는 값을 변환합니다.

사용 가능한 명령어

래더 다이어그램 및 평션 블록

DEG	RAD	TOD	FRD	TRN

ST(스트럭처드 텍스트)

DEG	RAD	TRN

실행할 작업	사용할 명령어
라디안을 도 단위로 변환합니다.	DEG
도 단위를 라디안으로 변환합니다.	RAD
정수 값을 BCD 값으로 변환합니다.	TOD
BCD 값을 정수 값으로 변환합니다.	FRD
값의 소수 부분을 제거합니다.	TRN

데이터 유형을 혼합할 수 있지만 정확도 손실 및 반올림 에러가 발생할 수 있으며 명령어를 실행하는 데 시간이 더 걸립니다. 결과가 잘렸는지 확인하려면 S:V 비트를 확인하십시오.

볼드체 데이터 유형은 최적의 데이터 유형을 나타냅니다. 명령어의 모든 피연산자가 동일한 최적의 데이터 유형(일반적으로 DINT 또는 REAL)을 사용하는 경우 명령어는 가장 적은 메모리를 사용하면서 보다 빠르게 실행됩니다.

수학 변환 명령어는 렁-입력-조건이 참인 경우 명령어가 스캔될 때마다 한 번 실행됩니다. 명령어를 한 번만 평가하려면 ONS 명령어를 사용하여 변환 명령어를 트리거하십시오.

추가 참조

계산/연산 명령어 페이지의 411

비교 명령어 페이지의 329

비트 명령어 페이지의 81

ASCII 문자열 명령어 페이지의 907

ASCII 변환 명령어 페이지의 927

BCD 로 변환(TOD)

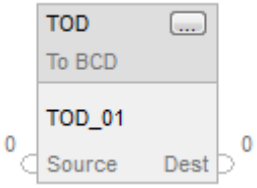
이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

TOD 명령어는 십진 값(0 ≤ Source ≤ 99,999,999)을 BCD 값으로 변환하고 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램

평션 블록



ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT	즉시 태그	BCD 로 변환할 값 0 ≤ Source ≤ 99,999,999
Destination	SINT INT DINT	태그	결과를 저장하는 태그

펑션 블록

피연산자	유형	형식	설명
TOD tag	FBD_CONVERT	구조	TOD 구조

FBD_CONVERT 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 해제된 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 설정 상태입니다.
Source	DINT	변환 명령어에 대한 입력입. 유효값 = 모든 정수

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	출력 활성화.
Dest	DINT	변환 명령어의 결과. 이 출력에 대해 연산 상태 플래그가 설정됩니다.

설명

BCD 는 4 비트 이진 표기법의 개별 십진수(0 ~ 9)로 표시되는 이진화 십진수 시스템입니다.

Source	Destination	대상 유형
음수 소스 < 0	0	
소스 > 99,999,999	16#9999_9999	DINT
소스 > 99,999,999	16#9999	INT
소스 > 99,999,999	16#99	SINT

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, *연산 상태 플래그* 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A.
렁-입력-조건이 거짓임	N/A.
렁-입력-조건이 참임	컨트롤러는 Source 를 BCD 로 변환하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A.

평션 블록

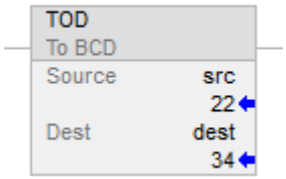
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

예

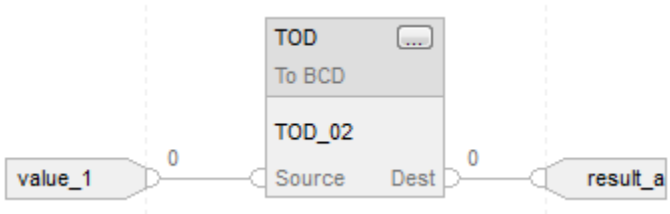
예 1

TOD 명령어는 value_1 을 BCD 값으로 변환하고 결과를 result_a 에 놓습니다.

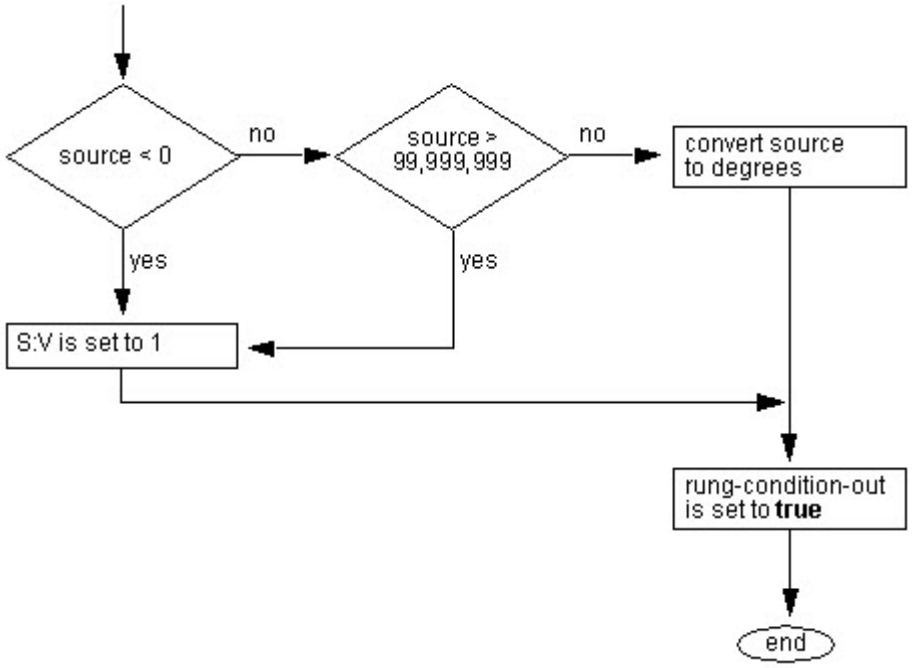
래더 다이어그램



평션 블록



TOD 흐름도(참)



추가 참조

계산 명령어 페이지의 411

공통 속성 페이지의 963

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

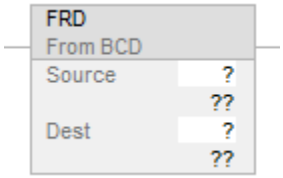
정수로 변환(FRD)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

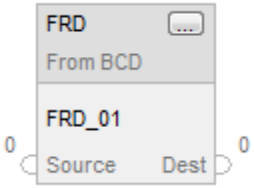
FRD 명령어는 BCD 값(Source)을 십진 값으로 변환하고 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT	즉시 태그	십진수로 변환할 값
Destination	SINT INT DINT	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

평션 블록

피연산자	유형	형식	설명
FRD tag	FBD_CONVERT	구조	FRD 구조

FBD_CONVERT 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
Source	DINT	변환 명령어에 대한 입력입. 유효값 = 모든 정수

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었는지를 나타냅니다.
Dest	DINT	변환 명령어의 결과.

설명

FRD 명령어는 BCD 값(Source)을 십진 값으로 변환하고 결과를 Destination 에 저장합니다.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, *연산 상태 플래그* 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

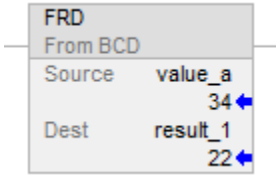
조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	컨트롤러는 Source 를 십진 값으로 변환하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

평션 블록

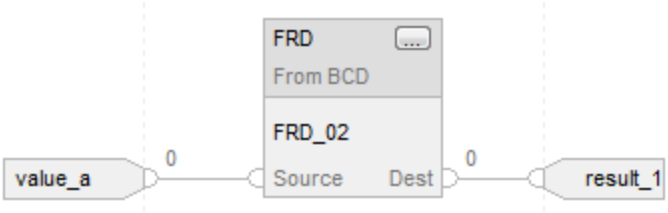
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

예

래더 다이어그램



평션 블록



추가 참조

계산 명령어 페이지의 411

공통 속성 페이지의 963

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

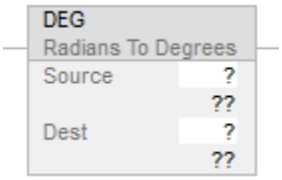
도(DEG)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

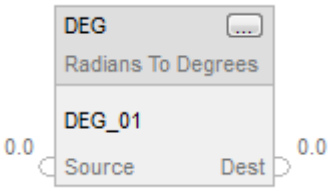
DEG 명령어는 Source(라디안 단위)를 도로 변환하고 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

dest := DEG(source);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Source	SINT INT DINT REAL	즉시 태그	도로 변환할 값
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

DEG 를 함수로 사용합니다. ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

평션 블록

피연산자	유형	형식	설명
DEG tag	FBD_MATH_ADVANCED	구조	DEG 구조

FBD_MATH_ADVANCED 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
Source	REAL	변환 명령어에 대한 입력.

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	명령어가 활성화되었는지를 나타냅니다.
Dest	REAL	변환 명령어의 결과.

설명

DEG 명령어는 다음 알고리즘을 사용합니다.

Source*180/pi = Source*57.29578

연산 상태 플래그에 영향

컨트롤러	영향을 받는 연산 상태 플래그
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, *연산 상태 플래그* 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

마이너 폴트 발생 조건:	폴트 유형	폴트 코드
오버플로가 감지됨	4	4

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	컨트롤러는 Source 를 라디안으로 변환하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

평션 블록

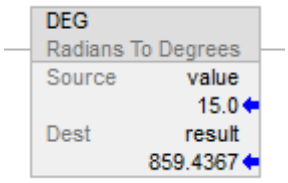
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

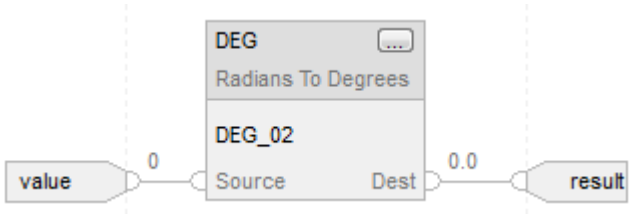
조건/상태	취해진 조치
사전 스캔	N/A
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	N/A

예

래더 다이어그램



평션 블록



ST(스트럭쳐드 텍스트)

result := DEG(value);

추가 참조

고급 연산 명령어 페이지의 821

공통 속성 페이지의 963

연산 상태 플래그 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

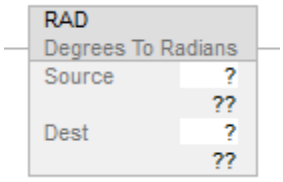
라디안(RAD)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

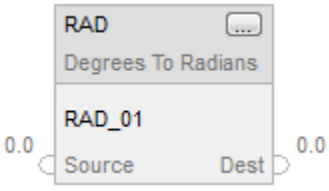
RAD 명령어는 Source(도 단위)를 라디안으로 변환하고 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

dest := RAD(source);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
소스	SINT INT DINT REAL	즉시 태그	라디안으로 변환할 값
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그

ST(스트럭처드 텍스트)

RAD 를 함수로 사용합니다. ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

펑션 블록

피연산자	유형	형식	설명
RAD tag	FBD_MATH_ADVANCED	구조	FRD 구조

FBD_MATH_ADVANCED 구조

입력 파라미터	데이터 유형	설명
활성화 입력	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
소스	REAL	변환 명령어에 대한 입력.

출력 파라미터	데이터 유형	설명
활성화 출력(EnableOut)	BOOL	출력 활성화.
Dest	REAL	변환 명령어의 결과.

연산 상태 플래그에 영향

컨트롤러	연산 상태 플래그에 영향
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, *연산 상태 플래그* 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	컨트롤러는 Source 를 라디안으로 변환하고 결과를 Destination 에 놓습니다.
사후 스캔	N/A

평션 블록

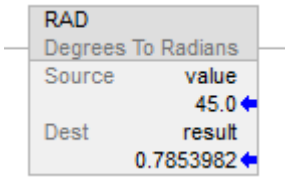
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

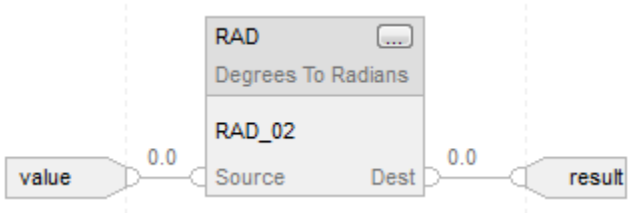
조건/상태	취해진 조치
사전 스캔	N/A
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	N/A

예

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

result := RAD(value);

추가 참조

ST(스트럭처드 텍스트) 구문 페이지의 997

공통 특성 페이지의 963

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

고급 연산 명령어 페이지의 821

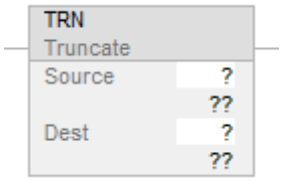
자르기(TRN)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

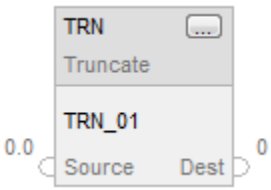
TRN 명령어는 Source 의 분수 부분을 제거하고(잘라냄) 결과를 Destination 에 저장합니다.

사용 가능한 언어

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

dest := TRUNC(source);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램 및 펑션 블록은 TRN 을 명령어로 사용합니다. TRN 명령어를 래더 다이어그램에 사용하는 경우 Source 피연산자는 REAL 태그 또는 즉시 값을 수락하기만 하며 대상은 REAL, DINT, SINT 및 INT 일 수 있습니다. 그러나 펑션 블록의 경우 대상은 DINT 만 가능합니다.

ST(스트럭처드 텍스트)는 TRUNC 를 연산자로 사용합니다. TRUNC 연산자의 경우 Source 피연산자는 REAL, SINT, INT 및 DINT 를 수락할 수 있습니다. 그러나 대상은 DINT 만 수락할 수 있습니다.

TRUNC 를 CPT 같은 명령어 식 안에 사용하면 TRUNC 를 연산자로 취합니다. Source 피연산자는 SINT, INT, DINT 및 REAL 같은 모든 정수 유형일 수 있습니다.

래더 다이어그램

피연산자	유형	형식	설명
Source*	REAL	즉시 태그	잘라낼 값
Destination	SINT INT DINT REAL	태그	결과를 저장하는 태그
데이터 변환: SINT 및 INT 태그는 부호 확장됩니다.			

펑션 블록

피연산자	유형	형식	설명
TRN tag	FBD_TRUNCATE	구조	TRN 구조

FBD_TRUNCATE 구조

입력 파라미터	데이터 유형	설명
EnableIn	BOOL	활성화 입력. 거짓일 경우 명령어가 실행되지 않고 출력이 업데이트되지 않습니다. 기본값은 참입니다.
Source	REAL	변환 명령어에 대한 입력. 입력은 DINT, SINT 및 INT 도 취합니다. 그러나 정수 유형은 먼저 REAL 유형으로 변환됩니다. SINT 또는 INT 를 REAL 로 변환하면 데이터 정밀도가 제대로 유지합니다. 그러나 DINT 를 REAL 로 변환하면 데이터 정밀도가 손실될 수 있습니다. 두 데이터 유형 모두 32 비트 데이터를 저장하지만 REAL 유형은 32 비트 중 일부를 지수 값을 저장하는 데 사용합니다. 정밀도를 상실한 경우 컨트롤러는 DINT 의 최하위 비트를 참조합니다.

출력 파라미터	데이터 유형	설명
EnableOut	BOOL	출력 활성화. Dest 가 오버플로하면 거짓으로 해제되고 그렇지 않으면 참으로 설정됩니다.
Dest	DINT	변환 명령어의 결과.

ST(스트럭처드 텍스트)

TRUNC 를 함수로 사용합니다. 이 함수는 소스를 잘라내고 정수 결과를 반환합니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

피연산자	유형	형식	설명
Source	REAL DINT SINT INT	즉시 태그	변환 명령어에 대한 입력.

설명

잘라내기는 값을 반올림하지 않으며, 대신에 분수가 아닌 부분은 분수 부분의 값과 상관없이 동일하게 유지합니다.

내부 연산을 오버플로할 수 있는 큰 실수를 잘라내면 0 값 대신에 값을 반환합니다.

TRN 을 래더 다이어그램 식의 연산자로 사용할 수 있으며, TRUNC 를 ST(스트럭처드 텍스트) 문의 연산자로 사용할 수 있습니다.

연산 상태 플래그에 영향

컨트롤러	영향을 받는 연산 상태 플래그
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	조건부, *연산 상태 플래그* 참조하십시오.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	예

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	렁은 거짓으로 설정됩니다.
렁-입력-조건이 거짓임	N/A.
렁-입력-조건이 참임	컨트롤러는 Source 의 분수 부분을 제거하고 결과를 Destination 에 놓습니다. 렁-입력-조건은 참으로 설정됩니다.
사후 스캔	렁은 거짓으로 설정됩니다.

평션 블록

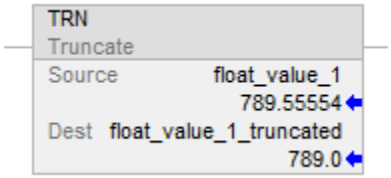
조건/상태	취해진 조치
사전 스캔	N/A
Tag.EnableIn 이 거짓임.	EnableOut 이 거짓으로 해제됩니다.
Tag.EnableIn 이 참임	EnableOut 이 참으로 설정됩니다. 블록이 오버플로를 생성하는 경우 EnableOut 이 거짓으로 해제됩니다.
명령어 최초 스캔	N/A
명령어 최초 실행	N/A
사후 스캔	N/A

ST(스트럭처드 텍스트)

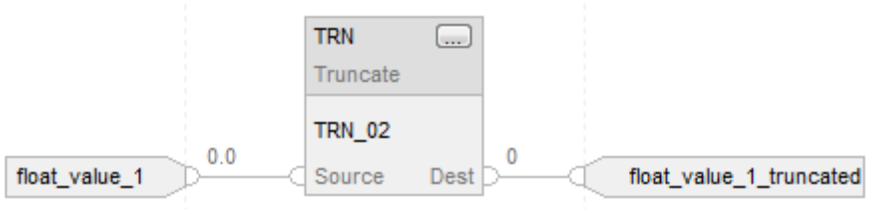
조건/상태	취해진 조치
사전 스캔	래더 다이어그램의 사전 스캔을 참조하십시오.
정상 실행	래더 다이어그램의 렁-입력-조건이 참으로 설정되는 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 확인하십시오.

예

래더 다이어그램



평션 블록



ST(스트럭처드 텍스트)

```
float_value_1_truncated := TRUNC(float_value_1);
```

추가 참조

ST(스트럭처드 텍스트) 구문 페이지의 997

고급 연산 명령어 페이지의 821

공통 속성 페이지의 963

연산 상태 플래그 페이지의 963

데이터 변환 페이지의 967

ASCII 시리얼 포트 명령어

ASCII 시리얼 포트 명령어

ASCII 문자를 읽고 쓰려면 ASCII 시리얼 포트 명령어를 사용하십시오.

중요: ASCII 시리얼 포트 명령어를 사용하려면 컨트롤러의 시리얼 포트를 구성해야합니다. 자세한 내용은 LOGIX 5000 Controllers Common Procedures Manual(발행 번호 1756-PM001)을 참조하십시오.

팁: ASCII 시리얼 포트 명령어(AWT, AWA, ARD, ARL, ABL, ACB, AHL, ACL)는 시리얼 포트가 없는 컨트롤러를 사용하는 프로젝트에 사용할 수 없습니다.

사용 가능한 명령어

래더 다이어그램과 ST(스트럭처드 텍스트)

ABL	ACB	ACL	AHL	ARD	ARL	AWA	AWT

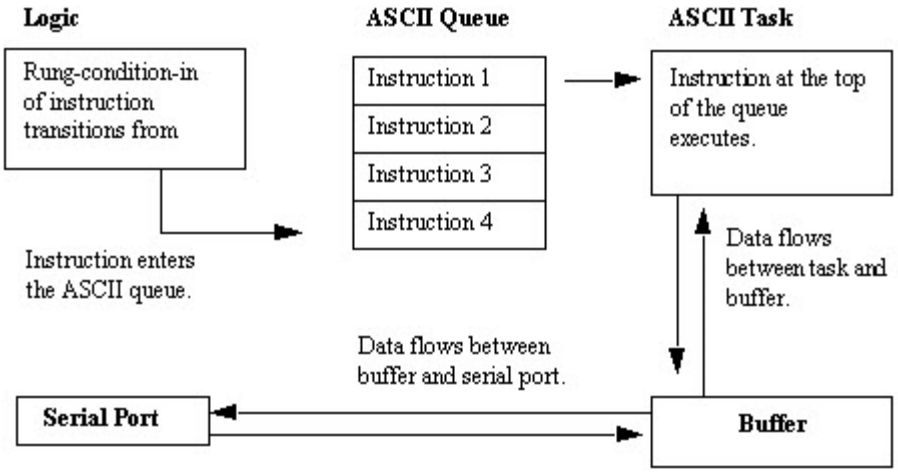
평션 블록

사용할 수 없음

실행할 작업:	사용할 명령어
종료 문자가 포함된 데이터를 확인합니다.	ABL
버퍼를 읽기 전에 필요한 문자 수를 확인합니다.	ACB
버퍼를 지웁니다. 예를 들어, 시작시 버퍼에서 오래된 데이터를 제거하거나 버퍼를 장치와 동기화합니다. 현재 실행 중이거나 큐에 있는 ASCII 시리얼 포트 명령어를 지웁니다.	ACL

시리얼 포트 제어 라인의 상태를 가져옵니다. 예를 들어, 모뎀이 끊어지도록 합니다. DTR 신호를 켜거나 끕니다. RTS 신호를 켜거나 끕니다.	AHL
고정된 수의 문자를 읽습니다. 예를 들어, 매 전송마다 동일한 수의 문자를 보내는 장치에서 데이터를 읽습니다.	ARD
첫 번째 종료 문자 집합을 포함하여 해당 문자까지 다양한 수의 문자를 읽습니다. 예를 들어, 매 전송마다 다른 수의 문자를 보내는 장치에서 데이터를 읽습니다.	ARL
문자를 보내고 하나 또는 두 개의 문자를 자동으로 추가하여 데이터의 끝을 표시합니다. 예를 들어, 항상 동일한 종료 문자를 사용하는 메세지를 보냅니다.	AWA
문자를 보냅니다. 예를 들어, 다양한 종료 문자를 사용하는 메세지를 보냅니다.	AWT

ASCII 시리얼 포트 명령어는 로직 스캔과 비동기적으로 실행됩니다.

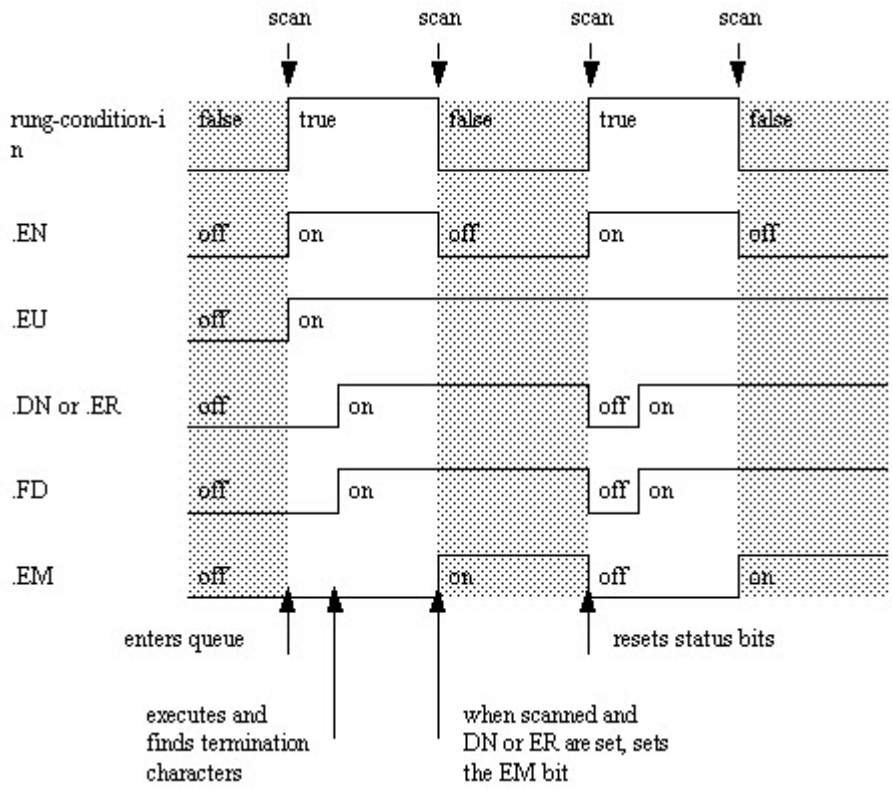


ACL 명령어를 제외한 각 ASCII 명령어는 SERIAL_PORT_CONTROL 구조를 사용합니다. SerialPort Control 피연산자는

- 명령어의 실행을 제어합니다.

- 명령어에 대한 상태 정보를 제공합니다. ASCII 명령어는 로직 스캔에 비동기적으로 실행됩니다.

SerialPort Control 피연산자의 비트에서 상태 정보를 제공합니다.



추가 참조

문자열 형식 페이지의 904

ASCII 에러 코드 페이지의 904

버퍼의 ASCII 문자(ACB)

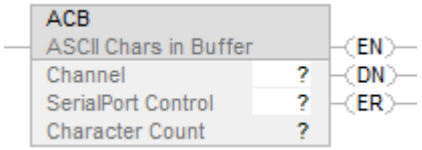
이 명령어는 Studio 5000 Logix Emulate 컨트롤러와만 호환됩니다.

ACB 명령어는 버퍼의 문자 수를 셉니다.

팁: ASCII 시리얼 포트 명령어(AWT, AWA, ARD, ARL, ABL, ACB, AHL, ACL)는 시리얼 포트가 없는 컨트롤러에 사용할 수 없습니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

ACB(Channel,SerialPortControl);

피연산자

래더 다이어그램

피연산자	유형	형식	설명
Channel	DINT	즉시 태그	0
SerialPort Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그
Character Count	DINT	즉시	0 실행하는 동안 첫 번째 종료 문자 집합을 포함하여 버퍼의 문자 수를 표시합니다.

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Channel	DINT	즉시 태그	0
SerialPort Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그
Character Count	DINT	즉시	0 실행하는 동안 첫 번째 종료 문자 집합을 포함하여 버퍼의 문자 수를 표시합니다.

피연산자 목록에 값을 포함시키는 대신 SERIAL_PORT_CONTROL 구조의 .POS 구성원에 액세스하여 Character Count 값을 지정할 수 있습니다.

ST(스트럭쳐드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

SERIAL_PORT_CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 명령어가 활성화되었음을 나타냅니다.
.EU	BOOL	큐는 명령어가 ASCII 큐에 들어갔음을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 완료된 시점을 나타내지만 로직 스캔과 비동기적입니다.
.RN	BOOL	실행 비트는 명령어가 실행 중임을 나타냅니다.
.EM	BOOL	빈 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 동기적입니다.
.ER	BOOL	에러 비트는 명령어가 실패한 경우(에러)를 나타냅니다.
.FD	BOOL	발견 비트는 명령어가 문자를 발견했다는 것을 나타냅니다.
.POS	DINT	위치는 첫 번째 종료 문자 집합까지 이를 포함하여 버퍼의 문자 수를 결정합니다.
.ERROR	DINT	에러는 에러의 원인을 식별하는 16 진수 값을 포함합니다.

설명

ACB 명령어는 버퍼의 문자 수를 셉니다.

ACB 명령어를 프로그래밍하려면 다음 지침을 따르십시오.

- 사용자 모드에 대한 컨트롤러의 시리얼 포트를 구성합니다.

이는 전환 명령어에 해당합니다.

- 래더 다이어그램에서 명령어가 실행해야 할 때마다 EnableIn 을 해제 상태에서 설정으로 전환합니다.
- ST(스트럭쳐드 텍스트)에서 명령어가 전환 시에만 실행하도록 조건을 설정합니다.

연산 상태 플래그

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

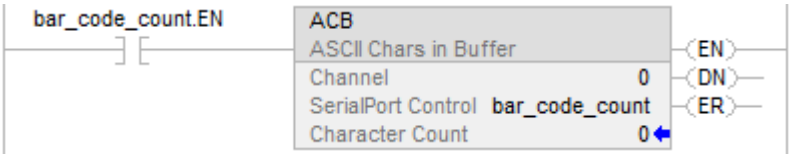
조건	래더 다이어그램 동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	EnableIn 이 해제 상태에서 설정으로 전환할 때 명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건	ST(스트럭처드 텍스트) 작업
사전 스캔	N/A
정상 실행	EnableIn 이 해제 상태에서 설정으로 전환할 때 명령어가 실행됩니다.
사후 스캔	N/A

예

래더 다이어그램



ST(스트럭처드 텍스트)

```
ACB(0,bar_code_count);
```

추가 참조

ASCII 시리얼 포트 명령어 페이지의 861

ST(스트럭쳐드 텍스트) 구문 페이지의 997

공통 속성 페이지의 963

ASCII 버퍼 지우기(ACL)

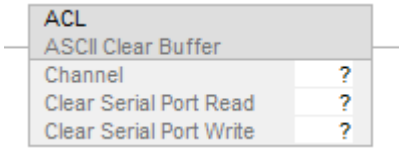
이 명령어는 Studio 5000 Logix Emulate 컨트롤러와만 호환됩니다.

ACL 명령어는 버퍼 및 ASCII 큐를 즉시 지웁니다.

팁: ASCII 시리얼 포트 명령어(AWT, AWA, ARD, ARL, ABL, ACB, AHL, ACL)는 시리얼 포트가 없는 컨트롤러에 사용할 수 없습니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

ACL(Channel,ClearSerialPortRead,ClearSerialPortWrite);

피연산자

래더 다이어그램

피연산자	유형	형식	설명
Channel	DINT	즉시 태그	0
Clear Serial Port Read	BOOL	즉시 태그	버퍼를 비우고 큐에서 ARD 및 ARL 명령어를 제거하려면 1 을 입력합니다.
Clear Serial Port Write	BOOL	즉시 태그	큐에서 AWA 및 AWT 명령어를 제거하려면 1 을 입력합니다.

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Channel	DINT	즉시 태그	0
Clear Serial Port Read	BOOL	즉시 태그	버퍼를 비우고 큐에서 ARD 및 ARL 명령어를 제거하려면 1 을 입력합니다.
Clear Serial Port Write	BOOL	즉시 태그	큐에서 AWA 및 AWT 명령어를 제거하려면 1 을 입력합니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

설명

ACL 명령어는 다음 동작 중 하나 또는 모두를 즉시 수행합니다.

- 버퍼를 지우거나 읽기 명령어의 ASCII 큐를 지웁니다.
- 쓰기 명령어의 ASCII 큐를 지웁니다. ACL 명령어를 프로그래밍하려면 다음 지침을 따르십시오.

컨트롤러의 시리얼 포트 구성:

응용 프로그램에서	그러면:
ARD 또는 ARL 명령어를 사용할 경우	사용자 모드를 선택합니다.
ARD 또는 ARL 명령어를 사용하지 않을 경우	시스템 또는 사용자 모드를 선택합니다.

명령어가 큐에서 제거되거나 중단되었는지 확인하려면 해당 명령어에 대해 다음을 검토하십시오.

- .ER 비트가 설정되었는지 여부
- .ERROR 구성원이 16#E 인지 여부

연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건	래더 다이어그램 동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭쳐드 텍스트)

조건	ST(스트럭쳐드 텍스트) 작업
사전 스캔	N/A
정상 실행	명령어가 지정된 명령어 및 버퍼를 지웁니다.
사후 스캔	N/A

예

래더 다이어그램

ST(스트럭쳐드 텍스트)

```
IF (osri_1.OutputBit THEN

ACL(0,0,1);

END_IF;
```

추가 참조

ASCII 시리얼 포트 명령어 페이지의 861

버퍼 라인에 대한 ASCII 테스트(ABL) 페이지의 888

버퍼의 ASCII 문자(ACB) 페이지의 863

ASCII 핸드셰이크 라인(AHL) 페이지의 870

ASCII 읽기(ARD) 페이지의 876

ASCII 읽기 라인(ARL) 페이지의 881

ASCII 쓰기 추가(AWA) 페이지의 898

ASCII 쓰기(AWT) 페이지의 892

ST(스트럭쳐드 텍스트) 구문 페이지의 997

공통 속성 페이지의 963

데이터 변환 페이지의 967

ASCII 핸드셰이크 라인(AHL)

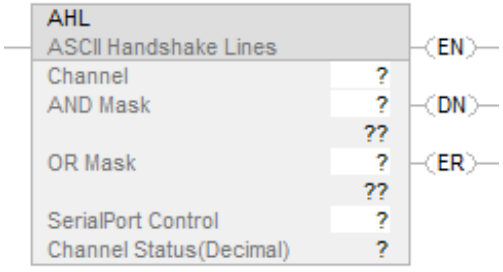
이 명령어는 Studio 5000 Logix Emulate 컨트롤러와만 호환됩니다.

AHL 명령어는 제어 라인의 상태를 파악하고 DTR 및 RTS 신호를 켜거나 끕니다.

팁: ASCII 시리얼 포트 명령어(AWT, AWA, ARD, ARL, ABL, ACB, AHL, ACL)는 시리얼 포트가 없는 컨트롤러에 사용할 수 없습니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

AHL(Channel,ANDMask,ORMask,SerialPortControl);

피연산자

래더 다이어그램

피연산자	유형	형식	설명	
Channel	DINT	즉시 태그	0	
ANDMask	DINT	즉시 태그	설명 참조하십시오.	
ORMask	DINT	즉시 태그		
SerialPort Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그	
Channel Status(Decimal)	DINT	즉시	0 실행 중 제어 라인의 상태를 표시합니다.	
			제어 라인의 상태:	**검사할 비트:**
			CTS	0
			RTS	1
			DSR	2
			DCD	3
			DTR	4
			XOFF 문자 수신	5

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	
Channel	DINT	즉시 태그	0	
ANDMask	DINT	즉시 태그	설명 참조하십시오.	
ORMask	DINT	즉시 태그		
SerialPort Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그	
Channel Status(Decimal)	DINT	즉시	0 실행 중 제어 라인의 상태를 표시합니다.	
			제어 라인의 상태:	**검사할 비트:**
			CTS	0
			RTS	1

			DSR	2
			DCD	3
			DTR	4
			XOFF 문자 수신	5

피연산자 목록에 값을 포함시키는 대신 SERIAL_PORT_CONTROL 구조의 .POS 구성원에 액세스하여 Channel Status 값을 지정할 수 있습니다.

ST(스트럭쳐드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

SERIAL_PORT_CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 명령어가 활성화되었음을 나타냅니다.
.EU	BOOL	큐 비트는 명령어가 ASCII 큐에 들어감을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 비동기적입니다.
.RN	BOOL	실행 비트는 명령어가 실행 중임을 나타냅니다.
.EM	BOOL	빈 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 동기적입니다.
.ER	BOOL	에러 비트는 명령어가 실패한 경우(에러)를 나타냅니다.
.FD	BOOL	발견된 비트는 이 명령어에 적용되지 않습니다.
.POS	DINT	위치는 첫 번째 종료 문자 집합까지 이를 포함하여 버퍼의 문자 수를 결정합니다.
.ERROR	DINT	에러는 에러의 원인을 식별하는 16 진수 값을 포함합니다.

설명

AHL 명령어는

- 시리얼 포트의 제어 라인 상태를 가져올 수 있습니다.
- DTR(Data Terminal Ready) 신호를 켜거나 끌 수 있습니다.
- RTS(Request to Send) 신호를 켜거나 끌 수 있습니다.

AHL 명령어를 프로그래밍하려면 다음 가이드라인을 따르십시오.

컨트롤러의 시리얼 포트 구성:

응용 프로그램에서	그러면:
ARD 또는 ARL 명령어를 사용할 경우	사용자 모드를 선택합니다.
ARD 또는 ARL 명령어를 사용하지 않을 경우	시스템 또는 사용자 모드를 선택합니다.

다음 표를 사용하여 ANDMask 및 ORMask 피연산자의 올바른 값을 선택하십시오.

DTR:	RTS:	다음 ANDMask 값을 입력:	그리고 다음 ORMask 값을 입력:
꺼짐	꺼짐	3	0
		켜짐	1
		변경되지 않음	1
켜짐	꺼짐	2	1
		켜짐	0
		변경되지 않음	0
변경되지 않음	꺼짐	2	0
		켜짐	0
		변경되지 않음	0

이는 전환 명령어에 해당합니다.

- 래더 다이어그램에서 명령어가 실행해야 할 때마다 EnableIn 을 해제 상태에서 설정으로 전환합니다.
- ST(스트럭쳐드 텍스트)에서 명령어가 전환 시에만 실행하도록 조건을 설정합니다.

연산 상태 플래그에 영향

아니요

폴트 조건

유형	코드	원인	복구 방법
4	57	시리얼 포트가 핸드셰이킹 없음으로 설정되었기 때문에 AHL 명령어가 실행되지 못합니다.	시리얼 포트의 제어 라인 설정을 변경합니다. 또는 AHL 명령어를 삭제합니다.

실행

래더 다이어그램

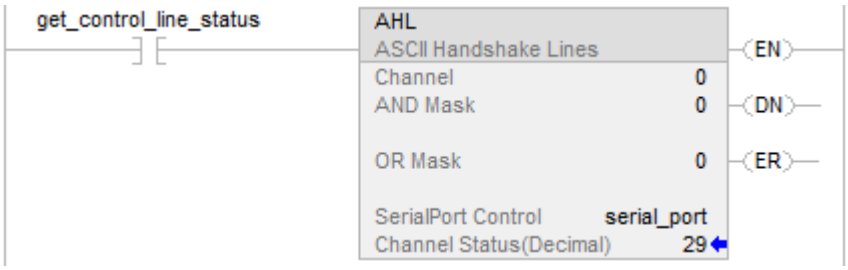
조건	래더 다이어그램 동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	렁 입력 조건이 해제에서 설정으로 전환될 때 명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭쳐드 텍스트)

조건	ST(스트럭쳐드 텍스트) 작업
사전 스캔	N/A
정상 실행	렁 입력 조건이 해제에서 설정으로 전환될 때 명령어가 실행됩니다.
사후 스캔	N/A

예

래더 다이어그램



ST(스트럭처드 텍스트)

```
osri_1.InputBit := get_control_line_status;

OSRI(osri_1);

IF (osri_1.OutputBit) THEN

AHL(0,0,0,serial_port);

END_IF;
```

추가 참조

ASCII 시리얼 포트 명령어 페이지의 861

버퍼 라인에 대한 ASCII 테스트(ABL) 페이지의 888

버퍼의 ASCII 문자(ACB) 페이지의 863

ASCII 버퍼 지우기(ACL) 페이지의 867

ASCII 읽기(ARD) 페이지의 876

ASCII 읽기 라인(ARL) 페이지의 881

ASCII 쓰기 추가(AWA) 페이지의 898

ASCII 쓰기(AWT) 페이지의 892

공통 속성 페이지의 963

ASCII 읽기(ARD)

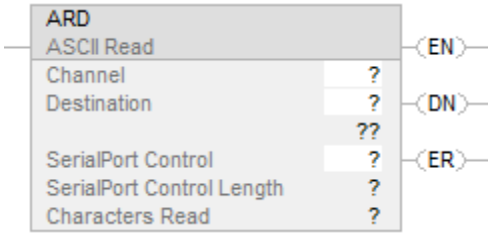
이 명령어는 Studio 5000 Logix Emulate 컨트롤러와만 호환됩니다.

ARD 명령어는 버퍼에서 문자를 제거하고 Destination 에 저장합니다.

팁: ASCII 시리얼 포트 명령어(AWT, AWA, ARD, ARL, ABL, ACB, AHL, ACL)는 시리얼 포트가 없는 컨트롤러에 사용할 수 없습니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

ARD(Channel,Destination,SerialPortControl);

피연산자

래더 다이어그램

피연산자	유형	형식	설명	참고:
Channel	DINT	즉시 태그	0	
Destination	문자열 유형 SINT INT DINT	태그	문자가 이동되는 태그(즉, 읽기): 문자열 유형의 경우 태그 이름을 입력합니다. SINT, INT 또는 DINT 배열의 경우 배열의 첫 번째 요소를 입력합니다.	문자를 비교, 변환 또는 조작하려면 문자열 유형 태그를 입력합니다. 문자열 유형은 기본 STRING 데이터 유형 사용자가 작성한 모든 새 문자열 유형
Serial Port Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그	
Serial Port Control Length	DINT	즉시	대상으로 이동하는 문자 수(읽기)	Serial Port Control Length 는 Destination 의 크기보다 작거나 같아야 합니다. Serial Port Control Length 를 Destination 의 크기와 같게 설정하려면 0 을 입력하십시오.
Characters Read	DINT	즉시	0	실행하는 동안 첫 번째 종료 문자 집합을 포함하여 버퍼의 문자 수를 표시합니다.

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Channel	DINT	즉시 태그	0	
Destination	문자열 유형 SINT INT DINT	태그	문자가 이동되는 태그(즉, 읽기): 문자열 유형의 경우 태그 이름을 입력합니다. SINT, INT 또는 DINT 배열의 경우 배열의 첫 번째 요소를 입력합니다.	문자를 비교, 변환 또는 조작하려면 문자열 유형 태그를 입력합니다. 문자열 유형은 기본 STRING 데이터 유형 사용자가 작성한 모든 새 문자열 유형
Serial Port Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그	
Serial Port Control Length	DINT	즉시	대상으로 이동하는 문자 수(읽기)	Serial Port Control Length 는 Destination 의 크기보다 작거나 같아야 합니다. Serial Port Control Length 를 Destination 의 크기와 같게 설정하려면 0 을 입력하십시오.
Characters Read	DINT	즉시	0	실행하는 동안 첫 번째 종료 문자 집합을 포함하여 버퍼의 문자 수를 표시합니다.

피연산자 목록에 값을 포함하는 대신 SERIAL_PORT_CONTROL 구조의 .LEN 및 .POS 구성원에 액세스하여 시리얼 포트 제어 길이 및 읽은 문자 수 값을 지정할 수 있습니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

SERIAL_PORT_CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 명령어가 활성화되었음을 나타냅니다.
.EU	BOOL	큐 비트는 명령어가 ASCII 큐에 들어갔음을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 비동기적입니다.
.RN	BOOL	실행 비트는 명령어가 실행 중임을 나타냅니다.
.EM	BOOL	빈 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 동기적입니다.
.ER	BOOL	에러 비트는 명령어가 실패한 경우(에러)를 나타냅니다.
.FD	BOOL	발견된 비트는 이 명령어에 적용되지 않습니다.
.LEN	DINT	길이는 대상으로 이동할 문자 수(즉, 읽기)를 나타냅니다.
.POS	DINT	위치는 읽은 문자 수를 표시합니다.
.ERROR	DINT	에러는 에러의 원인을 식별하는 16 진수 값을 포함합니다.

설명

ARD 명령어는 버퍼에서 지정된 수의 문자를 제거하고 대상에 저장합니다.

- ARD 명령어는 지정된 수의 문자(시리얼 포트 제어 길이 피연산자)가 제거될 때까지 계속 실행됩니다.
- ARD 명령어가 실행되는 동안 다른 ASCII 시리얼 포트 명령어는 실행되지 않습니다.

ARD 명령어를 프로그래밍하려면 다음 가이드라인을 따르십시오.

1. 사용자 모드에 대한 컨트롤러의 시리얼 포트를 구성합니다.
2. ACB 명령어의 결과를 사용하여 ARD 명령어를 트리거합니다. 이렇게 하면 ARD 명령어가 필요한 문자 수를 대기하는 동안 큐를 보류하지 않습니다. 자세한 내용은 아래 ARD 예제를 참조하십시오.
3. 이는 전환 명령어에 해당합니다.
   래더 다이어그램에서 명령을 실행해야 할 때마다 EnableIn 을 해제 상태에서 설정으로 전환합니다.
   ST(스트럭처드 텍스트)에서는 명령어가 전환 시에만 실행되도록 조건을 지정합니다.

4. 명령어가 완료될 때 후속 동작을 트리거하려면 .EM 비트를 검사하십시오.

연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

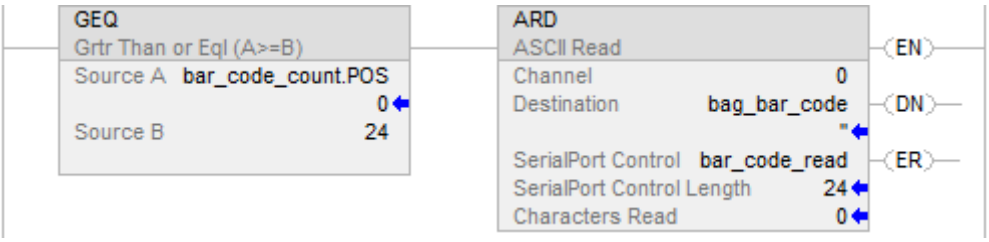
조건	래더 다이어그램 동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다. EnableIn 이 해제에서 설정으로 전환됩니다.
사후 스캔	N/A

ST(스트럭쳐드 텍스트)

조건	ST(스트럭쳐드 텍스트) 작업
사전 스캔	N/A
정상 실행	명령어가 실행됩니다. EnableIn 이 해제에서 설정으로 전환됩니다.
사후 스캔	N/A

예

래더 다이어그램



ST(스트럭쳐드 텍스트)

```
ACB(o,bar_code_count);

IF bar_code_count.POS >= 24 THEN

bar_code_read.LEN := 24;

ARD(0,bag_bar_code,bar_code_read);

END_IF;
```

추가 참조

ASCII 시리얼 포트 명령어 페이지의 861

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

ASCII 읽기 라인(ARL)

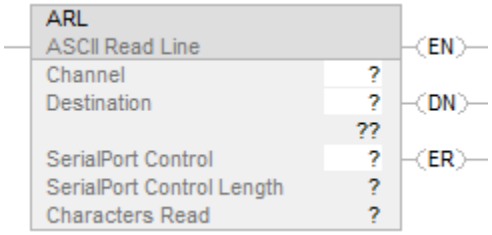
이 명령어는 Studio 5000 Logix Emulate 컨트롤러와만 호환됩니다.

ARL 명령어는 버퍼에서 문자를 제거하고 대상 링크에 저장합니다.

팁: ASCII 시리얼 포트 명령어(AWT, AWA, ARD, ARL, ABL, ACB, AHL, ACL)는 시리얼 포트가 없는 컨트롤러에 사용할 수 없습니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

ARL(Channel,Destination,SerialPortControl);

피연산자

래더 다이어그램

피연산자	유형	형식	설명	참고:
Channel	DINT	즉시 태그	0	
Destination	문자열 유형 SINT INT DINT	태그	문자가 이동되는 태그(즉, 읽기) 문자열 유형의 경우 태그 이름을 입력합니다. SINT, INT 또는 DINT 배열의 경우 배열의 첫 번째 요소를 입력합니다.	문자를 비교, 변환 또는 조작하려면 문자열 유형 태그를 입력합니다. 문자열 유형은 기본 STRING 데이터 유형 사용자가 작성한 모든 새 문자열 유형
SerialPort Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그	

Serial Port Control Length	DINT	즉시	종료 문자가 없는 경우 읽을 최대 문자 수.	모든 메세지에 포함될 최대 문자 수를 입력합니다(즉, 종료 문자가 없는 경우 읽기를 중지할 시점). 예를 들어, 메세지의 길이가 3 ~ 6 자인 경우 6 을 입력합니다. Serial Port Control Length 는 Destination 의 크기보다 작거나 같아야 합니다. Serial Port Control Length 를 Destination 의 크기와 같게 설정하려면 0 을 입력하십시오.
Characters Read	DINT	즉시	0	실행 중 읽은 문자 수를 표시합니다.

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Channel	DINT	즉시 태그	0	
Destination	문자열 유형 SINT INT DINT	태그	문자가 이동되는 태그(즉, 읽기) 문자열 유형의 경우 태그 이름을 입력합니다. SINT, INT 또는 DINT 배열의 경우 배열의 첫 번째 요소를 입력합니다.	문자를 비교, 변환 또는 조작하려면 문자열 유형 태그를 입력합니다. 문자열 유형은 기본 STRING 데이터 유형 사용자가 작성한 모든 새 문자열 유형
SerialPort Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그	

Serial Port Control Length	DINT	즉시	종료 문자가 없는 경우 읽을 최대 문자 수.	모든 메세지에 포함될 최대 문자 수를 입력합니다(즉, 종료 문자가 없는 경우 읽기를 중지할 시점). 예를 들어, 메세지의 길이가 3 ~ 6 자인 경우 6 을 입력합니다. Serial Port Control Length 는 Destination 의 크기보다 작거나 같아야 합니다. Serial Port Control Length 를 Destination 의 크기와 같게 설정하려면 0 을 입력하십시오.
Characters Read	DINT	즉시	0	실행 중 읽은 문자 수를 표시합니다.

피연산자 목록에 값을 포함하는 대신 SERIAL_PORT_CONTROL 구조의 .LEN 및 .POS 구성원에 액세스하여 Serial Port Control Length 및 Characters Read 값을 지정합니다.

ST(스트럭쳐드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭쳐드 텍스트) 구문*을 참조하십시오.

SERIAL_PORT_CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 명령어가 활성화되었음을 나타냅니다.
.EU	BOOL	큐 비트는 명령어가 ASCII 큐에 들어갔음을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 비동기적입니다.
.RN	BOOL	실행 비트는 명령어가 실행 중임을 나타냅니다.
.EM	BOOL	빈 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 동기적입니다.
.ER	BOOL	에러 비트는 명령어가 실패한 경우(에러)를 나타냅니다.
.FD	BOOL	발견된 비트는 이 명령어에 적용되지 않습니다.
.LEN	DINT	길이는 대상으로 이동할 최대 문자 수를 나타냅니다(즉, 종료 문자가 없는 경우 읽기를 중지할 시점).

.POS	DINT	위치는 읽은 문자 수를 표시합니다.
.ERROR	DINT	에러는 에러의 원인을 식별하는 16 진수 값을 포함합니다.

설명

ARL 명령어는 다음과 같이 버퍼에서 문자를 제거하고 대상에 저장합니다.

- ARL 명령어는 다음 중 하나를 제거할 때까지 계속 실행됩니다.
  - 종료 문자의 첫 번째 집합
  - 지정된 문자 수(문자열 길이 피연산자)

ARL 명령어가 실행되는 동안 다른 ASCII 명령어는 실행되지 않습니다. ARL 명령어를 프로그래밍하려면 다음 지침을 따르십시오.

1. 사용자 모드에 대해 컨트롤러의 시리얼 포트를 구성하고 종료 문자로 사용할 문자를 정의하십시오.
2. ABL 명령어의 결과를 사용하여 ARL 명령어를 트리거합니다. 이렇게 하면 ARL 명령어가 종료 문자를 대기하는 동안 큐를 보류하지 않습니다. 자세한 내용은 아래 ARL 예를 참조하십시오.
3. 이는 전환 명령어에 해당합니다. 래더 다이어그램에서 명령을 실행해야 할 때마다 EnableIn 을 해제 상태에서 설정으로 전환합니다. ST(스트럭쳐드 텍스트)에서는 명령어가 전환 시에만 실행되도록 조건을 지정합니다.
4. 명령어가 완료될 때 후속 동작을 트리거하려면 .EM 비트를 검사하십시오.

연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건	래더 다이어그램 동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다. EnableIn 이 해제에서 설정으로 전환됩니다.
사후 스캔	N/A

ST(스트럭쳐드 텍스트)

조건	ST(스트럭쳐드 텍스트) 작업
사전 스캔	N/A
정상 실행	명령어가 실행됩니다. EnableIn 이 해제에서 설정으로 전환됩니다.
사후 스캔	N/A

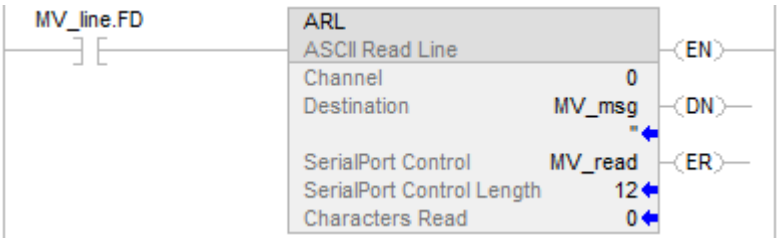
예

MessageView 터미널에서 메세지에 대한 버퍼를 지속적으로 테스트합니다. 각 메세지는 캐리지 리턴($r)으로 끝나기 때문에 캐리지 리턴은 컨트롤러 속성(Controller Properties) 대화 상자의 사용자 프로토콜(User Protocol) 탭에서 종료 문자로 구성됩니다.

ABL 이 캐리지 리턴을 찾으면 .FD 비트를 설정합니다. ABL 명령어가 캐리지 리턴을 찾으면(MV_line.FD 가 설정됨), 컨트롤러가 완전한 메세지를 수신한 것입니다.

ARL 명령어는 버퍼의 문자를 캐리지 리턴까지 제거하고 이를 문자열 유인 MV_msg 태그의 DATA 구성원에 저장합니다.

래더 다이어그램



ST(스트럭처드 텍스트)

```
ABL(0,MV_line);

osri_1.InputBit :=MVLine.FD

OSRI(osri_1);

IF osri_1.OutputBit) THEN

mv_read.LEN := 12;

ARL(0,MV_msg,MV_read);

END_IF;
```

추가 참조

ASCII 시리얼 포트 명령어 페이지의 861

버퍼 라인에 대한 ASCII 테스트(ABL) 페이지의 888

버퍼의 ASCII 문자(ACB) 페이지의 863

ASCII 버퍼 지우기(ACL) 페이지의 867

ASCII 핸드셰이크 라인(AHL) 페이지의 870

ASCII 읽기(ARD) 페이지의 876

ASCII 쓰기 추가(AWA) 페이지의 898

ASCII 쓰기(AWT) 페이지의 892

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

버퍼 라인에 대한 ASCII 테스트(ABL)

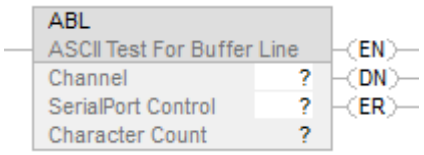
이 명령어는 Studio 5000 Logix Emulate 컨트롤러와만 호환됩니다.

ABL 명령어는 버퍼의 문자를 첫 번째 종료 문자까지 카운트합니다.

팁: ASCII 시리얼 포트 명령어(AWT, AWA, ARD, ARL, ABL, ACB, AHL, ACL)는 시리얼 포트가 없는 컨트롤러에 사용할 수 없습니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

ABL(Channel,SerialPortControl);

피연산자

래더 다이어그램

피연산자	유형	형식	설명
Channel	DINT	즉시	0
SerialPort Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그
Character Count	DINT	즉시	0 실행하는 동안 첫 번째 종료 문자 집합을 포함하여 버퍼의 문자 수를 표시합니다.

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Channel	DINT	즉시	0
SerialPort Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그
Character Count	DINT	즉시	0 실행하는 동안 첫 번째 종료 문자 집합을 포함하여 버퍼의 문자 수를 표시합니다.

문자 카운트 값은 SERIAL_PORT_CONTROL 구조의 .POS 구성원을 통해 액세스합니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

SERIAL_PORT_CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 명령어가 활성화되었음을 나타냅니다.
.EU	BOOL	큐 비트는 명령어가 ASCII 큐에 들어갔음을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 완료된 시점을 나타내지만 로직 스캔과 비동기적입니다.
.RN	BOOL	실행 비트는 명령어가 실행 중임을 나타냅니다.
.EM	BOOL	빈 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 동기적입니다.
.ER	BOOL	에러 비트는 명령어가 실패한 경우(에러)를 나타냅니다.
.FD	BOOL	발견된 비트는 명령어가 종료 문자를 찾았음을 나타냅니다.
.POS	DINT	위치는 첫 번째 종료 문자 집합까지 이를 포함하여 버퍼의 문자 수를 결정합니다. 명령어는 종료 문자를 찾은 후에만 이 번호를 반환합니다.
.ERROR	DINT	에러는 에러의 원인을 식별하는 16 진수 값을 포함합니다.

설명

ABL 명령어는 버퍼에서 첫 번째 종료 문자 집합을 검색합니다. 명령어가 종료 문자를 찾으면

- .FD 비트가 설정.

- 버퍼의 문자를 첫 번째 종료 문자 집합까지 카운트합니다.

컨트롤러 속성(Controller Properties) 대화 상자의 **사용자 프로토콜**(User Protocol) 탭에서 명령어가 종료 문자로 간주하는 ASCII 문자를 정의합니다.

ABL 명령어를 프로그래밍하려면 다음 가이드라인을 따르십시오.

- 사용자 모드에 대해 컨트롤러의 시리얼 포트를 구성하고 종료 문자로 사용할 문자를 정의하십시오.

이는 전환 명령어에 해당합니다.

- 래더 다이어그램에서 명령어가 실행해야 할 때마다 EnableIn 을 해제 상태에서 설정으로 전환합니다.
- ST(스트럭쳐드 텍스트)에서 명령어가 전환 시에만 실행하도록 조건을 설정합니다.

연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

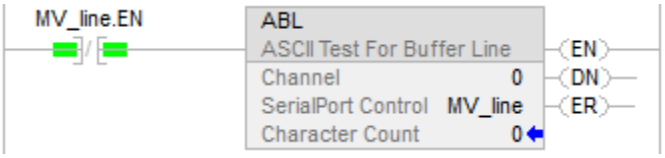
조건	래더 다이어그램 동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다. EnableIn 이 해제에서 설정으로 전환됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건	ST(스트럭처드 텍스트) 작업
사전 스캔	N/A
정상 실행	명령어가 실행됩니다. EnableIn 이 해제에서 설정으로 전환됩니다.
사후 스캔	N/A

예

래더 다이어그램



ST(스트럭처드 텍스트)

```
ABL(0,MV_line);
```

추가 참조

ASCII 시리얼 포트 명령어 페이지의 861

버퍼의 ASCII 문자(ACB) 페이지의 863

ASCII 버퍼 지우기(ACL) 페이지의 867

ASCII 핸드셰이크 라인(AHL) 페이지의 870

ASCII 읽기(ARD) 페이지의 876

ASCII 읽기 라인(ARL) 페이지의 881

ASCII 쓰기 추가(AWA) 페이지의 898

ASCII 쓰기(AWT) 페이지의 892

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

ASCII 쓰기(AWT)

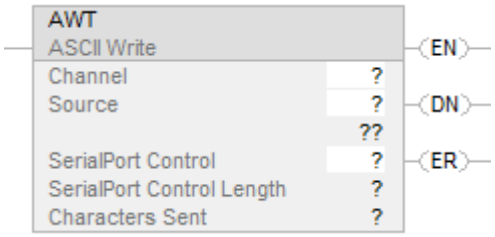
이 명령어는 Studio 5000 Logix Emulate 컨트롤러와만 호환됩니다.

AWT 명령어는 Source 배열의 문자를 시리얼 장치로 보냅니다.

팁: ASCII 시리얼 포트 명령어(AWT, AWA, ARD, ARL, ABL, ACB, AHL, ACL)는 시리얼 포트가 없는 컨트롤러에 사용할 수 없습니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

AWT(Channel,Source,SerialPortControl);

피연산자

래더 다이어그램

피연산자	유형	형식	설명	참고:
Channel	DINT	즉시 태그	0	
Source	문자열 유형 SINT INT DINT	태그	보낼 문자가 포함된 태그 문자열 유형의 경우 태그 이름을 입력합니다. SINT, INT 또는 DINT 배열의 경우 배열의 첫 번째 요소를 입력합니다.	문자를 비교, 변환 또는 조작하려면 문자열 유형 태그를 입력합니다. 문자열 유형은 기본 STRING 데이터 유형 사용자가 작성한 모든 새 문자열 유형
Serial Port Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그	

Serial Port Control Length	DINT	즉시	보낼 문자 수	Serial Port Control Length 는 Source 의 크기보다 작거나 같아야 합니다. Serial Port Control Length 를 Source 의 문자 수와 같게 설정하려면 0 을 입력하십시오.
Characters Sent	DINT	즉시	0	실행 중, 보낸 문자 수를 표시합니다.

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Channel	DINT	즉시 태그	0	
Source	문자열 유형 SINT INT DINT	태그	보낼 문자가 포함된 태그 문자열 유형의 경우 태그 이름을 입력합니다. SINT, INT 또는 DINT 배열의 경우 배열의 첫 번째 요소를 입력합니다.	문자를 비교, 변환 또는 조작하려면 문자열 유형 태그를 입력합니다. 문자열 유형은 기본 STRING 데이터 유형 사용자가 작성한 모든 새 문자열 유형
Serial Port Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그	
Serial Port Control Length	DINT	즉시	보낼 문자 수	Serial Port Control Length 는 Source 의 크기보다 작거나 같아야 합니다. Serial Port Control Length 를 Source 의 문자 수와 같게 설정하려면 0 을 입력하십시오.
Characters Sent	DINT	즉시	0	실행 중, 보낸 문자 수를 표시합니다.

피연산자 목록에 값을 포함하는 대신 SERIAL_PORT_CONTROL 구조의 .LEN 및 .POS 구성원에 액세스하여 Serial Port Control Length 및 Characters Sent 값을 지정할 수 있습니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

SERIAL_PORT_CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 명령어가 활성화되었음을 나타냅니다.
.EU	BOOL	큐 비트는 명령어가 ASCII 큐에 들어갔음을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 비동기적입니다.
.RN	BOOL	실행 비트는 명령어가 실행 중임을 나타냅니다.
.EM	BOOL	빈 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 동기적입니다.
.ER	BOOL	에러 비트는 명령어가 실패한 경우(에러)를 나타냅니다.
.FD	BOOL	발견된 비트는 이 명령어에 적용되지 않습니다.
.LEN	DINT	보낼 문자 수를 나타내는 길이.
.POS	DINT	보낸 문자 수를 표시하는 위치.
.ERROR	DINT	에러는 에러의 원인을 식별하는 16 진수 값을 포함합니다.

설명

AWT 명령어는 소스 태그의 지정된 문자 수(즉, 시리얼 포트 제어 길이)를 컨트롤러의 시리얼 포트에 연결된 장치로 전송합니다.

AWT 명령어를 프로그래밍하려면 다음 지침을 따르십시오.

1. 컨트롤러의 시리얼 포트 구성:

응용 프로그램에서	그러면:
ARD 또는 ARL 명령어를 사용할 경우	사용자 모드를 선택합니다.
ARD 또는 ARL 명령어를 사용하지 않을 경우	시스템 또는 사용자 모드를 선택합니다.

2. 이는 전환 명령어에 해당합니다. 래더 다이어그램에서 명령어가 실행해야 할 때마다 EnableIn 을 해제 상태에서 설정으로 전환합니다. ST(스트럭쳐드 텍스트)에서 명령어가 전환 시에만 실행하도록 조건을 설정합니다.

3. 명령어가 실행될 때마다 항상 같은 수의 문자를 보냅니까?

다음의 경우:	그러면:
예	Serial Port Control Length 에서 보낼 문자 수를 입력하십시오.
아니요	명령어를 실행하기 전에 소스 태그의 LEN 구성원을 Serial Port Control 태그의 LEN 구성원으로 이동합니다. 아래 예 2 를 참조하십시오.

연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건	래더 다이어그램 동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다. EnableIn 이 해제에서 설정으로 전환됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

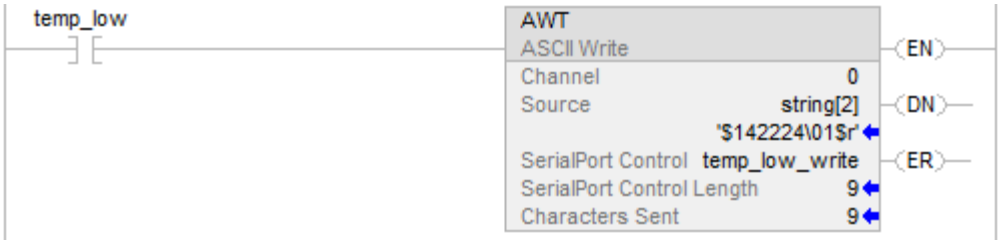
조건	ST(스트럭처드 텍스트) 작업
사전 스캔	N/A
정상 실행	명령어가 실행됩니다. EnableIn 이 해제에서 설정으로 전환됩니다.
사후 스캔	N/A

예

예 1

온도가 하한에 도달하면(즉, temp_low 가 켜짐) AWT 명령어는 컨트롤러의 시리얼 포트에 연결된 MessageView 터미널에 메세지를 전송합니다. 메세지에는 문자열 유형인 string[2] 태그의 DATA 구성원의 9 자가 포함됩니다. ($14 는 한 문자로 간주되며 Ctrl-T 문자의 16 진수 코드입니다.) 마지막 문자는 메세지 끝을 표시하는 캐리지 리턴($r)입니다.

래더 다이어그램



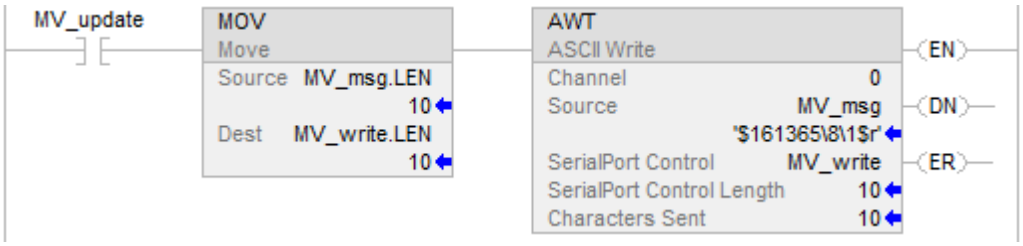
ST(스트럭처드 텍스트)

```
osri_1.InputBit := temp_low;

OSRI(osri_1);

IF (osri_1.OutputBit) THEN

temp_low_write.LEN := 9;

AWT(0.string[2],temp_low_write);

END_IF;
```

예 2

MV_update 가 켜지면 AWT 명령어는 Mv_msg 에 있는 문자를 보냅니다. MV_msg 의 문자 수는 다양하기 때문에 렁은 먼저 문자열(MV_msg.LEN)의 길이를 AWT 명령어의 Serial Port Control Length 로 이동합니다(MV_write.LEN). (MV_msg 에서 $16 은 한 문자로 간주되며 Ctrl-V 문자의 16 진수 코드입니다.)

래더 다이어그램



ST(스트럭쳐드 텍스트)

```
osri_1.InputBit := MV_update;

OSRI(osri_1);

IF (osri_1.OutputBit) THEN

MV_write.LEN := Mv_msg.LEN;

AWT(0.MV_msg,MV_write);

END_IF;
```

추가 참조

ASCII 시리얼 포트 명령어 페이지의 861

버퍼 라인에 대한 ASCII 테스트(ABL) 페이지의 888

버퍼의 ASCII 문자(ACB) 페이지의 863

ASCII 버퍼 지우기(ACL) 페이지의 867

ASCII 핸드셰이크 라인(AHL) 페이지의 870

ASCII 읽기(ARD) 페이지의 876

ASCII 읽기 라인(ARL) 페이지의 881

ASCII 쓰기 추가(AWA) 페이지의 898

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

ASCII 쓰기 추가(AWA)

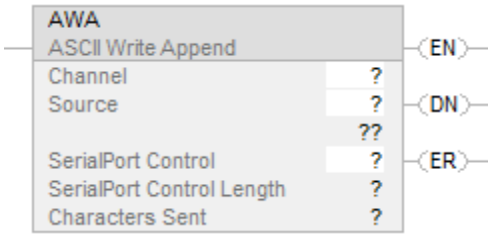
이 명령어는 Studio 5000 Logix Emulate 컨트롤러와만 호환됩니다.

AWA 명령어는 Source 배열의 문자를 시리얼 장치에 보내고 하나 또는 두 개의 미리 정의된 문자를 첨부합니다.

팁: ASCII 시리얼 포트 명령어(AWT, AWA, ARD, ARL, ABL, ACB, AHL, ACL)는 시리얼 포트가 없는 컨트롤러에 사용할 수 없습니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

AWA(Channel,Source,SerialPortControl);

피연산자

래더 다이어그램

피연산자	유형	형식	설명	참고:
Channel	DINT	즉시 태그	0	
Source	문자열 유형 SINT INT DINT	태그	보낼 문자가 포함된 태그 문자열 유형의 경우 태그 이름을 입력합니다. SINT, INT 또는 DINT 배열의 경우 배열의 첫 번째 요소를 입력합니다.	문자를 비교, 변환 또는 조작하려면 문자열 유형 태그를 입력합니다. 문자열 유형은 기본 STRING 데이터 유형 사용자가 작성한 모든 새 문자열 유형
Serial Port Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그	

Serial Port Control Length	DINT	즉시	보낼 문자 수	Serial Port Control Length 는 Source 의 크기보다 작거나 같아야 합니다. Serial Port Control Length 를 Source 의 문자 수와 같게 설정하려면 0 을 입력하십시오.
Characters Sent	DINT	즉시	0	실행 중, 보낸 문자 수를 표시합니다.

ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Channel	DINT	즉시 태그	0	
Source	문자열 유형 SINT INT DINT	태그	보낼 문자가 포함된 태그 문자열 유형의 경우 태그 이름을 입력합니다. SINT, INT 또는 DINT 배열의 경우 배열의 첫 번째 요소를 입력합니다.	문자를 비교, 변환 또는 조작하려면 문자열 유형 태그를 입력합니다. 문자열 유형은 기본 STRING 데이터 유형 사용자가 작성한 모든 새 문자열 유형
Serial Port Control	SERIAL_PORT_CONTROL	태그	작업을 제어하는 태그	
Serial Port Control Length	DINT	즉시	보낼 문자 수	Serial Port Control Length 는 Source 의 크기보다 작거나 같아야 합니다. Serial Port Control Length 를 Source 의 문자 수와 같게 설정하려면 0 을 입력하십시오.
Characters Sent	DINT	즉시	0	실행 중, 보낸 문자 수를 표시합니다.

피연산자 목록에 값을 포함하는 대신 SERIAL_PORT_CONTROL 구조의 .LEN 및 .POS 구성원에 액세스하여 Serial Port Control Length 및 Characters Sent 값을 지정할 수 있습니다.

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

SERIAL_PORT_CONTROL 구조

니모닉	데이터 유형	설명
.EN	BOOL	활성화 비트는 명령어가 활성화되었음을 나타냅니다.
.EU	BOOL	큐 비트는 명령어가 ASCII 큐에 들어갔음을 나타냅니다.
.DN	BOOL	완료 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 비동기적입니다.
.RN	BOOL	실행 비트는 명령어가 실행 중임을 나타냅니다.
.EM	BOOL	빈 비트는 명령어가 완료되었음을 나타내지만 로직 스캔과 동기적입니다.
.ER	BOOL	에러 비트는 명령어가 실패한 경우(에러)를 나타냅니다.
.FD	BOOL	발견된 비트는 이 명령어에 적용되지 않습니다.
.LEN	DINT	보낼 문자 수를 나타내는 길이.
.POS	DINT	보낸 문자 수를 표시하는 위치.
.ERROR	DINT	에러는 에러의 원인을 식별하는 16 진수 값을 포함합니다.

설명

AWA 명령어:

- Source 태그의 지정된 문자 수(즉, 시리얼 포트 제어 길이)를 컨트롤러의 시리얼 포트에 연결된 장치로 전송합니다.
- 컨트롤러 속성(Controller Porperties) 대화 상자의 사용자 프로토콜(User Protocol) 탭에 정의된 문자 하나 또는 두 개를 문자의 끝에 추가합니다(즉, 첨부).

AWA 명령어를 프로그래밍하려면 다음 지침을 따르십시오.

1. 컨트롤러의 시리얼 포트 구성:

응용 프로그램에서	그러면:
ARD 또는 ARL 명령어를 사용할 경우	사용자 모드를 선택합니다.
ARD 또는 ARL 명령어를 사용하지 않을 경우	시스템 또는 사용자 모드를 선택합니다.

2. 이는 전환 명령어에 해당합니다. 래더 다이어그램에서 명령어가 실행해야 할 때마다 EnableIn 을 해제 상태에서 설정으로 전환합니다.
   ST(스트럭쳐드 텍스트)에서 명령어가 전환 시에만 실행하도록 조건을 설정합니다.

3. 명령어가 실행될 때마다 항상 같은 수의 문자를 보냅니까?

다음의 경우:	그러면:
예	Serial Port Control Length 에서 보낼 문자 수를 입력하십시오.
아니요	명령어를 실행하기 전에 소스 태그의 LEN 구성원을 Serial Port Control 태그의 LEN 구성원으로 이동합니다. (아래 예 2 를 참조하십시오)

연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건	래더 다이어그램 동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다. EnableIn 이 해제에서 설정으로 전환됩니다.
사후 스캔	N/A

ST(스트럭쳐드 텍스트)

조건	ST(스트럭쳐드 텍스트) 작업
사전 스캔	N/A
정상 실행	명령어가 실행됩니다. EnableIn 이 해제에서 설정으로 전환됩니다.
사후 스캔	N/A

예

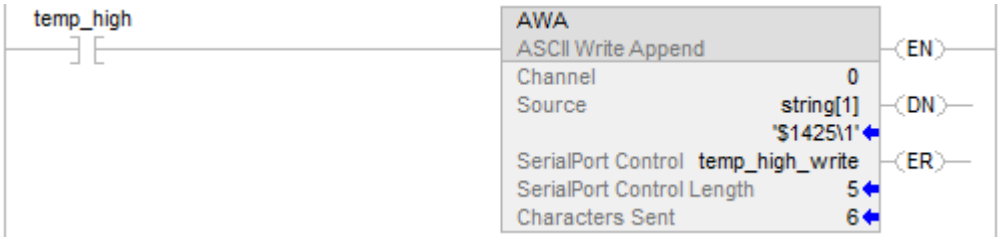
예 1

온도가 상한을 초과하면(temp_high 가 켜짐) AWA 명령어는 컨트롤러의 시리얼 포트에 연결된 MessageView 터미널에 메세지를 전송합니다.

메세지에는 문자열 유형인 string[1] 태그의 DATA 구성원의 5 자가 포함됩니다. ($14 는 한 문자로 간주되며 Ctrl-T 문자의 16 진수 코드입니다.)

이 명령어는 또한 컨트롤러 속성에 정의된 문자를 전송(첨부)합니다. 이 예에서 AWA 명령어는 메세지 끝을 표시하는 캐리지 리턴($0D)을 보냅니다.

래더 다이어그램



ST(스트럭처드 텍스트)

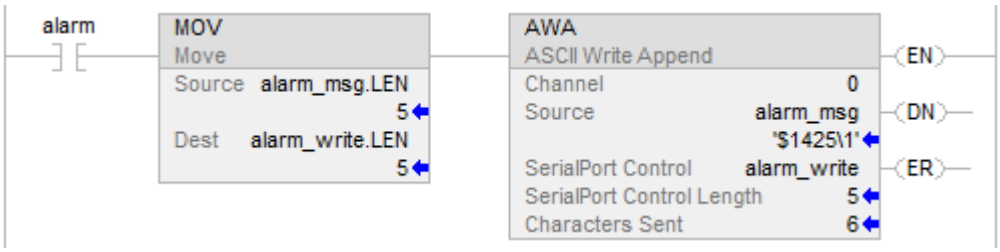
```
IF temp_high THEN

temp_high_write.LEN := 5;

AWA(o,string[1],temp_high_write);

temp_high := 0;

END_IF;
```

예 2

알람이 켜지면 AWA 명령어는 alarm_msg 에 지정된 수의 문자를 보내고 종료 문자를 추가합니다 alarm_msg 의 문자 수는 다양하기 때문에 렁은 먼저 문자열(alarm_msg.LEN)의 길이를

AWA 명령어의 Serial Port Control Length(alarm_write.LEN)로 이동합니다. alarm_msg 에서 $14 는 한 문자로 간주되며 Ctrl-T 문자의 16 진수 코드입니다.

래더 다이어그램



ST(스트럭쳐드 텍스트)

```
osri_1.InputBit := alarm;

OSRI(osri_1);

IF(osri_1.OutputBit) THEN

alarm_write.LEN := alarm_msg.LEN;

AWA(0,alarm_msg,alarm_write);

END_IF;
```

추가 참조

ASCII 시리얼 포트 명령어 페이지의 861

버퍼 라인에 대한 ASCII 테스트(ABL) 페이지의 888

버퍼의 ASCII 문자(ACB) 페이지의 863

ASCII 버퍼 지우기(ACL) 페이지의 867

ASCII 핸드셰이크 라인(AHL) 페이지의 870

ASCII 읽기(ARD) 페이지의 876

ASCII 읽기 라인(ARL) 페이지의 881

ASCII 쓰기(AWT) 페이지의 892

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

문자열 형식

태그에 문자열 형식 데이터 유형을 사용하는 ASCII 문자가 저장됩니다.

- 기본값 STRING 데이터 유형을 사용하여 최대 82 문자를 저장
- 더 적거나 많은 문자를 저장하는 새 문자열 형식을 만듦

새 문자열 형식을 작성하려면 LOGIX 5000 Controllers ASCII Strings Programming Manual (발행 번호 1756-PM013)을 참조하십시오.

각 문자열 형식은 다음 구성원을 포함합니다.

이름(Name)	데이터 유형	설명	참고
LEN	DINT	문자열의 문자 수	다음을 사용할 때마다 LEN 이 새 문자 카운트로 자동 업데이트됩니다. • 문자 입력을 위한 문자열 브라우저 • 문자열을 읽거나, 변환하거나 조작하는 명령어 LEN 은 현재 문자열의 길이를 표시합니다. DATA 구성원은 LEN 카운트에 포함되지 않는 추가적인 이전 문자를 포함할 수 있습니다.
DATA	SINT 배열	문자열의 ASCII 문자	문자열의 문자에 액세스하려면 태그의 이름에 따라 주소를 검색합니다. 예를 들어 string_1 태그의 문자에 액세스하려면 string_1 을 입력합니다. DATA 배열의 각 요소는 한 문자를 포함합니다. 그보다 적거나 많은 문자를 저장하는 새로운 문자열 형식을 작성하십시오.

추가 참조

문자열 리터럴 페이지의 1012

ASCII 에러 코드

ASCII 시리얼 포트 명령어를 실행하지 못하면 해당 SERIAL_PORT_CONTROL 구조의 ERROR 구성원에 다음과 같은 16 진수 에러 코드 중 하나가 포함됩니다.

16 진수 코드	의미:
16#2	모뎀이 오프라인 상태로 전환됐습니다.
16#3	통신 중에 CTS 신호가 손실됩니다.

16#4	시리얼 포트가 시스템 모드였습니다.
16#5	채널 구성 메뉴를 통해 채널 구성이 종료되었기 때문에 명령어를 보내거나 받을 수 없었습니다.
16#6	잘못된 파라미터가 ASCII 드라이버로 전달됩니다.
16#7	채널 구성 메뉴를 통해 채널 구성이 종료되었기 때문에 명령어를 보내거나 받을 수 없었습니다.
16#8	전송이 이미 진행 중입니다. 이 경우 진행 중인 명령어에 에러가 발생합니다.
16#9	요청된 ASCII 통신은 현재 채널 구성에서 지원되지 않습니다.
16#10	채널이 시스템 모드에 있는 동안 AHL 명령어를 실행하려고 시도했습니다.
16#A	명령어가 실행되기 전에 UL 비트가 설정됩니다. 이 경우 명령어의 실행이 중지됩니다.
16#B	이 명령어가 작동하도록 요청된 포트가 존재하지 않습니다.
16#C	컨트롤러가 실행 모드에서 프로그램 모드로 변경됩니다. 이 경우 ASCII 시리얼 포트 명령어의 실행을 중지하고 큐를 지웁니다
16#D	컨트롤러 속성(Controller Properties) 대화 상자의 사용자 프로토콜(User Protocol) 탭에서 버퍼 크기 또는 에코 모드 파라미터가 변경 및 적용됩니다. 이 경우 ASCII 시리얼 포트 명령어의 실행을 중지하고 큐를 지웁니다
16#E	ACL 명령어가 이러한 유형의 명령어를 실행, 중지 또는 제거했습니다.
16#F	시리얼 포트 구성이 사용자 모드에서 시스템 모드로 변경됩니다. 이 경우 ASCII 시리얼 포트 명령어의 실행을 중지하고 큐를 지웁니다
16#51	문자열 태그의 LEN 값이 문자열 태그의 DATA 크기보다 크거나 음수입니다.
16#54	Serial Port Control Length 가 버퍼의 크기보다 큽니다.
16#55	Serial Port Control Length 가 Source 또는 Destination 의 크기보다 크거나 음수입니다.

ASCII 문자열 명령어

ASCII 문자열 명령어

ASCII 문자열 명령어를 사용하여 ASCII 문자의 문자열을 수정 및 작성합니다.

사용 가능한 명령어

래더 다이어그램과 ST(스트럭처드 텍스트)

FIND	INSERT	MID	CONCAT	DELETE

평션 블록

사용할 수 없음

실행할 작업:	사용할 명령어:
문자열에 종료 문자 또는 구분 기호를 추가합니다.	CONCAT
문자열에서 문자를 삭제합니다(예: 문자열에서 머리글 또는 제어 문자 제거).	DELETE
하위 문자열의 시작 문자를 결정합니다.	FIND
문자열에 문자를 삽입	INSERT
문자열에서 문자를 추출합니다.	MID

다음 명령어를 사용하여 ASCII 문자를 비교하거나 변환할 수도 있습니다.

실행할 작업:	사용할 명령어:
문자열을 다른 문자열과 비교합니다.	CMP
문자가 특정 문자와 같은지 확인합니다.	EQU
문자가 특정 문자와 같지 않은지 확인합니다.	NEQ
문자가 특정 문자보다 크거나 같은지 확인합니다.	GEQ
문자가 특정 문자보다 큰지 확인합니다.	GRT

문자가 특정 문자보다 작거나 같은지 확인합니다.	LEQ
문자가 특정 문자보다 작은지 확인합니다.	LES
INT, DINT 또는 REAL 태그의 바이트를 다시 정렬합니다.	SWPB
문자열 배열에서 문자열을 찾습니다.	FSC
문자를 SINT, INT, DINT 또는 REAL 값으로 변환합니다.	STOD
문자를 REAL 값으로 변환합니다.	STOR
SINT, INT, DINT 또는 REAL 값을 ASCII 문자의 문자열로 변환합니다.	DTOS
REAL 값을 ASCII 문자의 문자열로 변환합니다.	RTOS

추가 참조

ASCII 에러 코드 페이지의 904

문자열 형식 페이지의 904

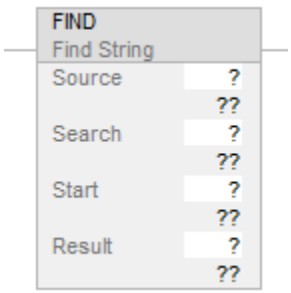
문자열 찾기(FIND)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

FIND 명령어는 다른 문자열 내에서 지정된 문자열의 시작 위치를 찾습니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

FIND(Source,Search,Start,Result);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Source	ANY_STRING	태그	검색이 수행되는 문자열	문자열 유형은 문자열의 최대 문자 길이가 82 자인 기본 STRING 데이터 유형. 문자열에 대해 구성 가능한 길이의 문자로 작성한 새로운 문자열 유형입니다.
Search	ANY_STRING	태그	찾을 문자열	
Start	SINT INT DINT	즉시 태그	Source 에서 검색을 시작할 위치	1 과 소스의 DATA 크기 사이의 숫자를 입력합니다.
Result	DINT SINT INT	태그	검색 문자열을 찾은 Source 내 위치	

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

설명

FIND 명령어는 Source 문자열에서 Search 문자열을 검색합니다. 명령어에서 Search 문자열을 찾으면 결과에 Source 문자열 내의 Search 문자열의 시작 위치가 표시됩니다. 그렇지 않으면 Result 가 0 입니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

마이너 폴트 발생 조건:	폴트 유형	폴트 코드
문자열 태그의 LEN 값이 문자열 태그의 DATA 크기보다 큽니다.	4	51
Start 값이 유효하지 않거나 Source 문자열이 비어 있습니다.	4	56

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건	동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

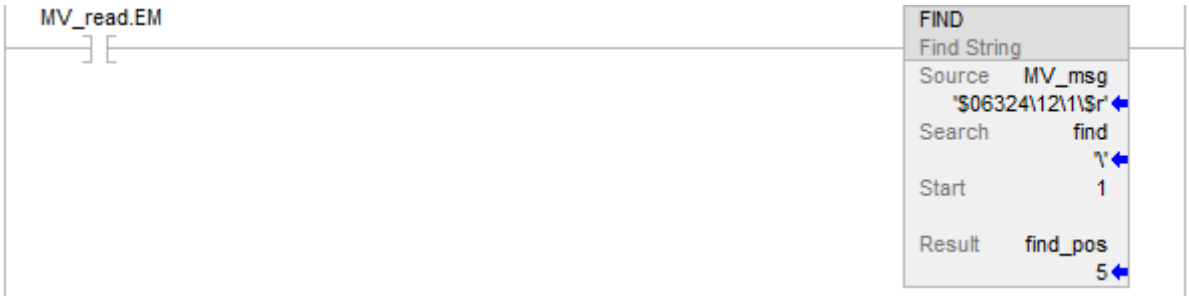
ST(스트럭처드 텍스트)

조건	동작
사전 스캔	래더 다이어그램 표의 사전 스캔 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표의 사후 스캔 참조하십시오.

예

MessageView 터미널의 메세지에는 여러 가지 정보가 들어 있습니다. 백슬래시 문자(\)는 각 정보를 구분합니다. 정보를 찾기 위해 FIND 명령어는 백슬래시 문자를 검색하고 find_pos 에 위치를 기록합니다.

래더 다이어그램



ST(스트럭쳐드 텍스트)

```
IF MV_read.EM THEN
  FIND(MV_msg,find,1,find_pos);
  MV_read.EM := 0;
END_IF;
```

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

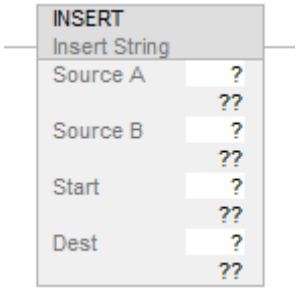
문자열 삽입(INSERT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

INSERT 명령어를 사용하여 문자열 내의 지정된 위치에 ASCII 문자를 추가합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

ST(스트럭처드 텍스트)

INSERT (SourceA,SourceB,Start,Dest);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오. INSERT 명령어는 다음 피연산자를 사용합니다.

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Source A	문자열 유형	태그	문자를 추가할 문자열	문자열 유형은 기본 STRING 데이터 유형 또는 생성한 새로운 문자열 유형입니다.
Source B	문자열 유형	태그	추가할 문자가 포함된 문자열	
Start	SINT DINT	즉시 태그	Source A 에서 문자가 추가되는 위치	1 과 소스의 DATA 크기 사이의 숫자를 입력합니다.
Destination	문자열 유형	태그	결과를 저장하는 문자열	

ST(스트럭쳐드 텍스트) 내 식의 구문에 대한 자세한 내용은 ST(스트럭쳐드 텍스트) 구문을 참조하십시오.

설명

INSERT 명령은 Source B 의 문자를 Source A 내의 지정된 위치에 추가하고 결과를 대상에 배치합니다.

- Start 는 Source A 에서 Source B 가 추가된 위치를 정의합니다.
- Source A 와 Destination 이 동일한 태그가 아니라면 Source A 는 변경되지 않습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

유형	코드	원인	복구 방법
4	51	문자열 태그의 LEN 값이 문자열 태그의 DATA 크기보다 큽니다.	1. 문자열 유형 태그의 LEN 구성원에 쓰고 있는 명령어가 없는지 확인합니다. 2. LEN 값에 문자열에 포함되는 문자의 수를 입력합니다.
4	56	Start 및 Quantity 값이 유효하지 않습니다.	Start 값이 1 과 Source 의 DATA 크기 사이에 있는지 확인하십시오.

실행

래더 다이어그램

조건	래더 다이어그램 동작
사전 스캔	렁-출력-조건이 거짓으로 설정됩니다.
렁-입력-조건이 거짓임	렁-출력-조건이 거짓으로 설정됩니다.
렁-입력-조건이 참임	명령어가 실행됩니다. 렁-출력-조건이 참으로 설정됩니다.
사후 스캔	렁-출력-조건이 거짓으로 설정됩니다.

실행

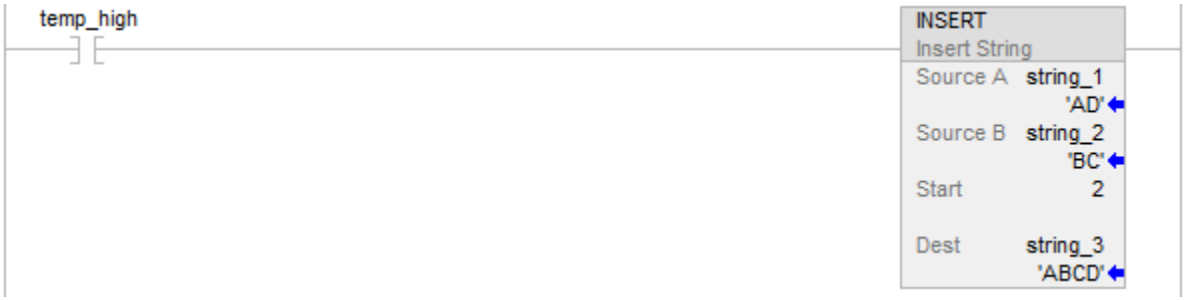
ST(스트럭처드 텍스트)

조건	동작
사전 스캔	래더 다이어그램 표의 사전 스캔 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표의 사후 스캔 참조하십시오.

예

temp_high 가 설정되면 INSERT 명령어는 *string_1* 의 2 위치에 *string_2* 의 문자를 추가하고 해당 결과를 *string_3* 에 저장합니다.

래더 다이어그램



ST(스트럭처드 텍스트)

```
IF temp_high THEN
  INSERT(string_1,string_2,2,string_3);
  temp_high := 0;
END_IF;
```

추가 참조

ASCII 문자열 명령어 페이지의 907

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

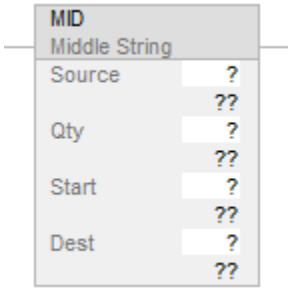
중간 문자열(MID)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

MID 명령어는 문자열에서 지정된 수의 ASCII 문자를 복사하여 다른 문자열에 저장합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

MID(Source,Qty,Start,Dest);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Source	ANY_STRING	태그	문자를 복사해올 문자열	문자열 유형은 문자열의 최대 문자 길이가 82 자인 기본 STRING 데이터 유형. 문자열에 대해 구성 가능한 길이의 문자로 작성한 새로운 문자열 유형입니다.
Quantity	SINT INT DINT	즉시 태그	복사할 문자 수	Start + Quantity 는 Source 의 길이에 1 을 더한 값보다 작거나 같아야 합니다.
Start	SINT INT DINT	즉시 태그	복사할 첫 번째 문자의 위치	1 과 소스의 DATA 크기 사이의 숫자를 입력합니다.
Destination	ANY_STRING	태그	문자를 복사할 대상 문자열	

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 ST(스트럭처드 텍스트) 구문을 참조하십시오.

설명

MID 명령어는 Source 에서 문자 그룹을 복사하고 Destination 에 결과를 저장합니다.

- Start 위치 및 Quantity 는 복사할 문자를 정의합니다.
- Source 와 Destination 이 동일한 Tag 가 아니라면 Source 는 변경되지 않습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

마이너 폴트 발생 조건:	폴트 유형	폴트 코드
소스 문자열 태그의 LEN 값이 Source 문자열 태그의 DATA 크기보다 큽니다.	4	51
출력 문자열의 길이가 대상 문자열 태그의 DATA 크기보다 큽니다.	4	52
Start 및 Quantity 값이 유효하지 않습니다.	4	56

실행

래더 다이어그램

조건	래더 다이어그램 동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

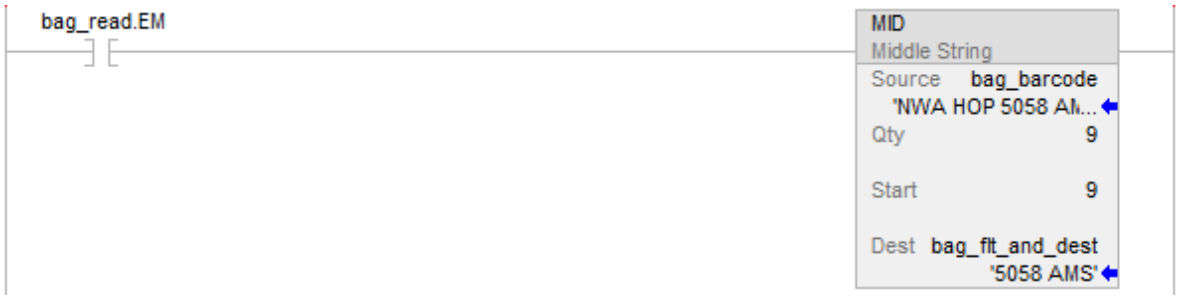
ST(스트럭처드 텍스트)

조건	동작
사전 스캔	래더 다이어그램 표의 사전 스캔 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표의 사후 스캔 참조하십시오.

예

공항의 수하물 처리 컨베이어에서 모든 가방에 바코드가 부여됩니다. 바코드의 9 번부터 17 번까지의 문자는 가방의 항공편 번호와 도착 공항을 나타냅니다. 바코드를 읽은 후(bag_read.EM 이 켜짐) MID 명령어는 항공편 번호와 도착 공항을 bag_flt_and_dest 문자열로 복사합니다. 후속 렁에서는 bag_flt_and_dest 를 사용하여 가방의 전송 경로를 결정합니다.

래더 다이어그램



ST(스트럭쳐드 텍스트)

```
IF bag_read.EM THEN
  MID(bag_barcode,9,9,bag_flt_and_dest);
  bag_read.EM := 0;
END_IF;
```

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

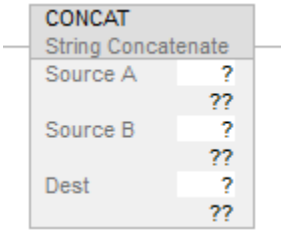
문자열 연결(CONCAT)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.;

CONCAT 명령어는 ASCII 문자를 문자열 끝에 추가합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

CONCAT(SourceA,SourceB,Dest);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환에 대한 자세한 내용은 공통 속성을 참조하십시오.

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Source A	ANY_STRING	태그	초기 문자가 포함된 태그	문자열 유형은 • 문자열의 최대 문자 길이가 82 자인 기본 STRING 데이터 유형. • 문자열에 대해 구성 가능한 길이의 문자로 작성한 새로운 문자열 유형입니다.
Source B	ANY_STRING	태그	종료 문자가 포함된 태그	
Destination	ANY_STRING	태그	결과를 저장하는 태그	

ST(스트럭처드 텍스트) 내에서 식의 구문에 대한 자세한 내용은 ST(스트럭처드 텍스트) 속성을 참조하십시오.

설명

CONCAT 명령어는 Source A 의 문자를 Source B 의 문자와 결합하여 결과를 Destination 에 저장합니다.

Source A 의 문자가 먼저 오고, 그 다음에 Source B 의 문자가 옵니다.

Source A 와 Destination 이 동일한 태그가 아니라면 Source A 는 변경되지 않습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

마이너 폴트 발생 조건:	폴트 유형	폴트 코드
문자열 태그의 LEN 값이 문자열 태그의 DATA 크기보다 큽니다.	4	51
Source A 와 Source B 의 합계 길이가 문자열 태그의 DATA 크기보다 큽니다.	4	51

배열 인덱스 폴트의 경우 배열을 통한 인덱스를 참조하십시오.

실행

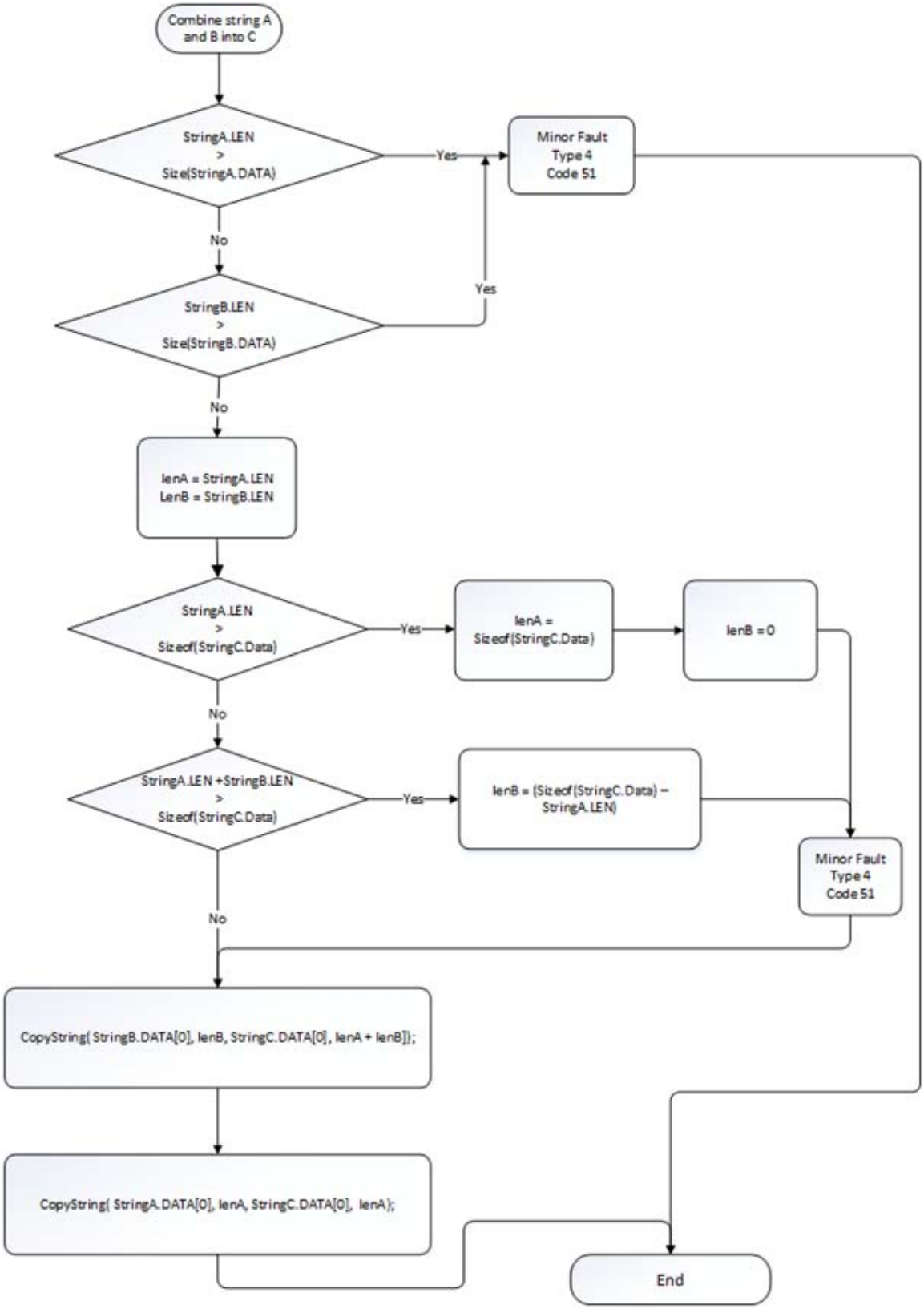
래더 다이어그램

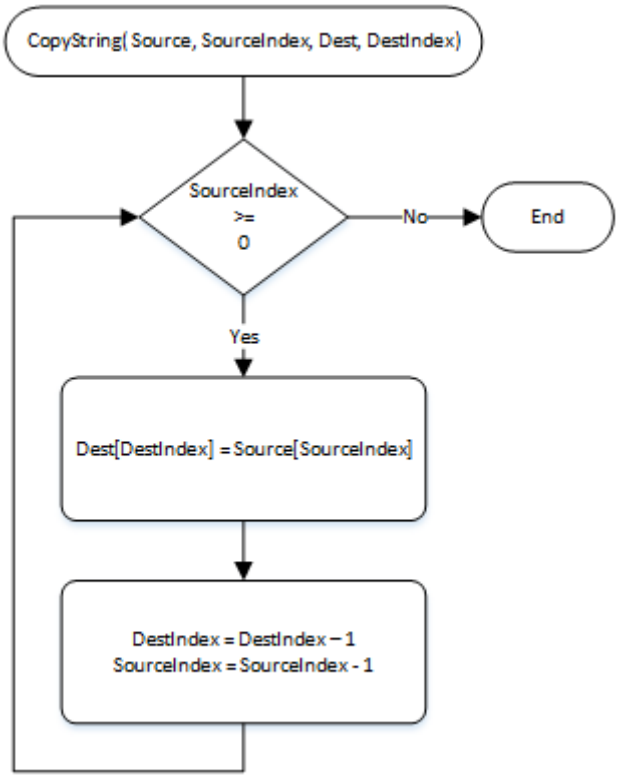
조건	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건	취해진 조치
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 확인하십시오.

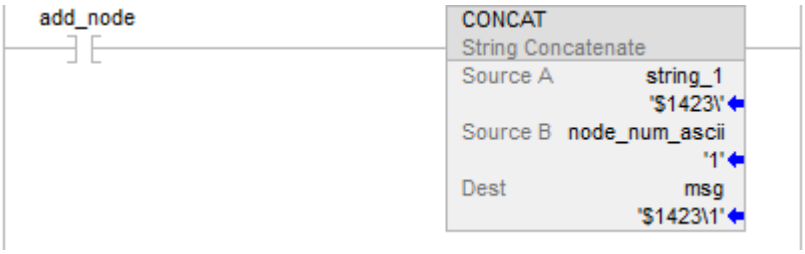
문자열 연결 흐름도


Combine string A and B into C
StringA.LEN > Size(StringA.DATA)
Yes
Minor Fault Type 4 Code 51
No
StringB.LEN > Size(StringB.DATA)
Yes
No
lenA = StringA.LEN
LenB = StringB.LEN
StringA.LEN > Sizeof(StringC.Data)
Yes
lenA = Sizeof(StringC.Data)
lenB = 0
No
StringA.LEN +StringB.LEN > Sizeof(StringC.Data)
Yes
lenB = (Sizeof(StringC.Data) – StringA.LEN)
Minor Fault Type 4 Code 51
No
CopyString(StringB.DATA[0], lenB, StringC.DATA[0], lenA + lenB]);
CopyString(StringA.DATA[0], lenA, StringC.DATA[0], lenA);
End



예

래더 다이어그램



ST(스트럭처드 텍스트)

CONCAT(string_1,string_2,msg);

추가 참조

공통 속성 페이지의 963

ST(스트럭처드 텍스트) 속성 페이지의 1028

데이터 변환 페이지의 967

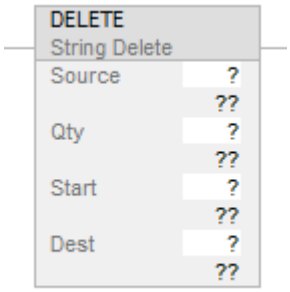
문자열 삭제(DELETE)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

DELETE 명령어는 문자열에서 ASCII 문자를 제거합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

DELETE(Source,Qty,Start,Dest);

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고
소스	ANY_STRING	태그	문자를 삭제할 문자열이 포함된 태그	문자열 유형은 문자열의 최대 문자 길이가 82 자인 기본 STRING 데이터 유형. 문자열에 대해 구성 가능한 길이의 문자로 작성한 새로운 문자열 유형입니다.
Quantity	SINT INT DINT	즉시 태그	삭제할 문자 수	Start + Quantity 는 Source 의 길이에 1 을 더한 값보다 작거나 같아야 합니다
시작(Search)	SINT INT DINT	즉시 태그	삭제할 첫 번째 문자의 위치	1 과 소스의 DATA 크기 사이의 숫자를 입력합니다.
Destination	문자열 유형	태그	결과를 저장할 태그	

ST(스트럭처드 텍스트) 내 식의 구문에 대한 자세한 내용은 ST(스트럭처드 텍스트) 구문을 참조하십시오.

설명

DELETE 명령어는 Source 에서 하나 이상의 문자를 삭제(제거)하고 Destination 에 나머지 문자를 저장합니다.

- Start 위치 및 Quantity 는 제거할 문자를 정의합니다.
- Source A 와 Destination 이 동일한 태그가 아니라면 Source A 는 변경되지 않습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

마이너 폴트 발생 조건:	폴트 유형	폴트 코드
소스 문자열 태그의 LEN 값이 Source 문자열 태그의 DATA 크기보다 큽니다.	4	51
출력 문자열의 길이가 대상 문자열 태그의 DATA 크기보다 큽니다.	4	52
Start 및 Quantity 값이 유효하지 않습니다.	4	56

피연산자 관련 폴트에 대해서는 공통 특성을 참조하십시오.

실행

래더 다이어그램

조건/상태	동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

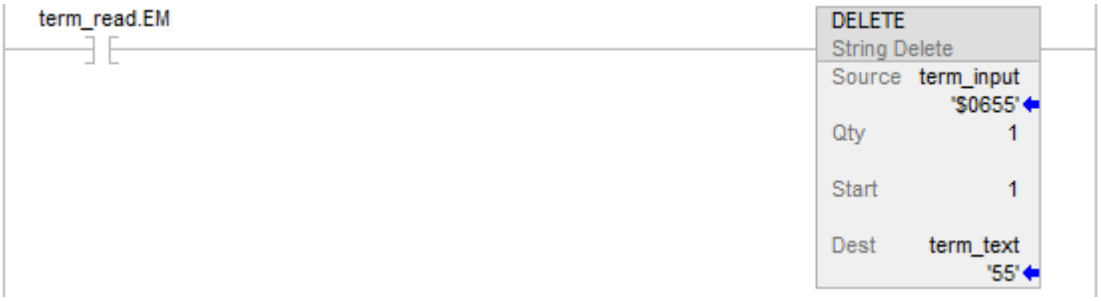
조건/상태	동작
사전 스캔	래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	래더 다이어그램 표의 렁-입력-조건이 참인 경우를 참조하십시오.
사후 스캔	래더 다이어그램 표에서 사후 스캔 섹션을 참조하십시오.

예제

터미널의 ASCII 정보에는 머리글 문자가 들어 있습니다. 컨트롤러가 데이터를 읽은 후(term_read.EM 이 켜짐), DELETE

명령어가 헤더 문자를 제거합니다. 그런 다음 컨트롤러는 메세지 텍스트를 사용하거나 이를 다른 장치로 전달할 수 있습니다.

래더 다이어그램



ST(스트럭쳐드 텍스트)

```
IF term_read.EM THEN
  DELETE(term_input,1,1,term_text);
  term_read.EM := 0;
END_IF;
```

추가 참조

공통 특성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

ASCII 변환 명령어

ASCII 변환 명령어

ASCII 변환 명령어를 사용하여 ASCII 문자의 문자열을 데이터로 변환하거나 그 반대로 변환합니다.

사용 가능한 명령어

래더 다이어그램과 ST(스트럭처드 텍스트)

STOD	STOR	RTOS	DTOS	LOWER	UPPER

평션 블록

사용할 수 없음

변환할 항목	사용할 명령어:
정수 값의 ASCII 표현을 SINT, INT, DINT 또는 REAL 값으로 변환할 경우(예: 저울 또는 다른 ASCII 장치의 값을 정수로 변환하여 로직에서 이를 사용할 경우)	STOD
부동 소수점 값의 ASCII 표현을 REAL 값으로 변환할 경우(예: 저울 또는 다른 ASCII 장치의 값을 REAL 값으로 변환하여 로직에서 이를 사용할 경우)	STOR
SINT, INT, DINT 또는 REAL 값을 ASCII 문자의 문자열로 변환할 경우(예: MessageView™ 터미널로 전송할 수 있도록 변수를 ASCII 문자열로 변환할 경우).	DTOS
REAL 값을 ASCII 문자의 문자열로 변환할 경우(예: MessageView 터미널로 전송할 수 있도록 변수를 ASCII 문자열로 변환할 경우).	RTOS

ASCII 문자의 문자열을 대문자로 변환할 경우(예: 배열에서 검색할 수 있도록 작업자가 입력한 항목을 모두 대문자로 변환할 경우).	UPPER
ASCII 문자의 문자열을 소문자로 변환할 경우(예: 배열에서 검색할 수 있도록 작업자가 입력한 항목을 모두 소문자로 변환할 경우).	LOWER

다음 명령어를 사용하여 ASCII 문자를 비교하거나 조작할 수도 있습니다.

실행할 작업:	**사용할 명령어:**
문자열의 끝에 문자를 추가	CONCAT
문자열에서 문자 삭제	DELETE
하위 문자열의 시작 문자를 결정합니다.	FIND
문자열에 문자를 삽입	INSERT
문자열에서 문자를 추출합니다.	MID
INT, DINT 또는 REAL 태그의 바이트를 다시 정렬합니다.	SWPB
문자열을 다른 문자열과 비교합니다.	CMP
문자가 특정 문자와 같은지 확인합니다.	EQU
문자가 특정 문자와 같지 않은지 확인합니다.	NEQ
문자가 특정 문자보다 크거나 같은지 확인합니다.	GEQ
문자가 특정 문자보다 큰지 확인합니다.	GRT
문자가 특정 문자보다 작거나 같은지 확인합니다.	LEQ
문자가 특정 문자보다 작은지 확인합니다.	LES
문자열 배열에서 문자열을 찾습니다.	FSC

추가 참조

ASCII 에러 코드 페이지의 904

문자열 형식 페이지의 904

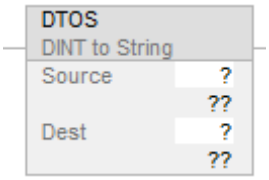
DINT -> 문자열(DTOS)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

DTOS 명령어는 값의 ASCII 표현을 생성합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

DTOS(Source,Dest);

피연산자

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Source	SINT INT DINT REAL	태그	값을 포함하는 태그	Source가 REAL일 경우 명령어에서 이를 DINT 값으로 변환합니다.
Destination	문자열 유형	태그	정수 값을 저장할 태그	문자열 유형은 • 기본 STRING 데이터 유형 • 사용자가 작성한 모든 새 문자열 유형

설명

DTOS 명령어는 Source 를 ASCII 문자의 문자열로 변환하고 Destination 에 결과를 저장합니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

유형	코드	원인	복구 방법
4	51	문자열 태그의 LEN 값이 문자열 태그의 DATA 크기보다 큽니다.	문자열 유형 태그의 LEN 구성원에 쓰고 있는 명령어가 없는지 확인합니다. LEN 값에 문자열에 포함되는 문자의 수를 입력합니다.
4	52	출력 문자열이 대상보다 큽니다.	출력 문자열에 대한 충분한 크기의 새 문자열 유형을 만듭니다. 새 문자열 유형을 대상의 데이터 유형으로 사용합니다.

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭쳐드 텍스트)

조건	동작
사전 스캔	이전 래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	이전 래더 다이어그램 표의 렁-입력-조건이 참을 참조하십시오.
사후 스캔	이전 래더 다이어그램 표의 사후 스캔을 참조하십시오.

예

temp_high 가 설정된 경우 DTOS 명령어는 msg_num 의 값을 ASCII 문자의 문자열로 변환하고 msg_num_ascii 에 결과를 저장합니다. 후속 렁에서는 msg_num_ascii 를 다른 문자열과 함께 삽입 또는 연결하여 디스플레이 터미널에 대한 완전한 메세지를 생성합니다.

래더 다이어그램

ST(스트럭쳐드 텍스트)

```
IF temp_high THEN
  DTOS(msg_num,msg_num_ascii);
  temp_high := 0;
END_IF;
```

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

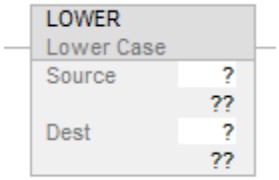
케이스 낮추기(LOWER)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

LOWER 명령어는 문자열의 알파벳 문자를 소문자로 변환합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

LOWER(Source,Dest);

피연산자

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Source	문자열	태그	소문자로 변환할 문자를 포함하는 태그
Destination	문자열	태그	소문자로 문자를 저장하는 태그

ST(스트럭처드 텍스트) 내에서 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트)*를 참조하십시오.

설명

LOWER 명령어는 Source 의 모든 문자를 소문자로 변환하고 Destination 에 결과를 저장합니다.

- ASCII 문자는 대소문자가 구분됩니다. 대문자 A($41)는 소문자 a($61)와 동일하지 않습니다.
- 작업자가 ASCII 문자를 직접 입력할 경우 비교하기 전에 문자를 모두 대문자 또는 소문자로 변환하십시오.

Source 문자열에서 문자가 아닌 문자는 변경되지 않습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

유형	코드	원인	복구 방법
4	51	문자열 태그의 LEN 값이 문자열 태그의 DATA 크기보다 큽니다.	문자열 유형 태그의 LEN 구성원에 쓰고 있는 명령어가 없는지 확인합니다. LEN 값에 문자열에 포함되는 문자의 수를 입력합니다.
4	52	출력 문자열이 대상보다 큽니다.	출력 문자열에 대한 충분한 크기의 새 문자열 유형을 만듭니다. 새 문자열 유형을 대상의 데이터 유형으로 사용합니다.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭쳐드 텍스트)

조건	동작
사전 스캔	이전 래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	이전 래더 다이어그램 표의 렁-입력-조건이 참을 참조하십시오.
사후 스캔	이전 래더 다이어그램 표의 사후 스캔을 참조하십시오.

예

특정 항목에 대한 정보를 찾기 위해 작업자가 항목 번호를 ASCII 터미널에 입력합니다. 컨트롤러가 터미널에서 입력을 읽은 후에(terminal_read 가 설정 됨), LOWER 명령어는 item_number 의 문자를 모두 소문자로 변환하고 결과를 item_number_lower_case 에 저장합니다. 그런 다음 후속 렁은 배열에서 item_number_lower_case 의 문자와 일치하는 문자를 검색합니다.

래더 다이어그램

ST(스트럭쳐드 텍스트)

```
IF terminal_read THEN
  LOWER(item_number,item_number_lower_case);
  terminal_read := 0;
END_IF;
```

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

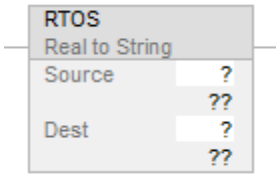
REAL -> 문자열(RTOS)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

RTOS 명령어는 REAL 값의 ASCII 표현을 생성합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

RTOS(Source,Dest);

피연산자

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Source	REAL	태그	REAL 값을 포함하는 태그	
Destination	문자열 유형	태그	ASCII 값을 저장할 태그	문자열 유형은 • 기본 STRING 데이터 유형 • 사용자가 작성한 모든 새 문자열 유형

식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

설명

RTOS 명령어는 Source 를 ASCII 문자의 문자열로 변환하고 Destination 에 결과를 저장합니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

유형	코드	원인	복구 방법
4	52	출력 문자열이 대상보다 큽니다.	출력 문자열에 대한 충분한 크기의 새 문자열 유형을 만듭니다. 새 문자열 유형을 대상의 데이터 유형으로 사용합니다.

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건	동작
사전 스캔	이전 래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	이전 래더 다이어그램 표의 렁-입력-조건이 참을 참조하십시오.
사후 스캔	이전 래더 다이어그램 표의 사후 스캔을 참조하십시오.

예

send_data 가 설정된 경우 RTOS 명령어는 data_1 의 값을 ASCII 문자의 문자열로 변환하고 data_1_ascii 에 결과를 저장합니다. 후속 렁에서는 data_1_ascii 를 다른 문자열과 함께 삽입 또는 연결하여 디스플레이 터미널에 대한 완전한 메세지를 생성합니다.

래더 다이어그램

ST(스트럭쳐드 텍스트)

```
IF send_data THEN

RTOS(data_1,data_1_ascii);

send_data:= 0;

END_IF;
```

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

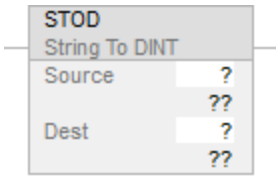
문자열 -> DINT(STOD)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

STOD 명령어는 정수의 ASCII 표현을 정수 또는 REAL 값으로 변환합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

STOD(Source,Dest);

피연산자

명령어 내에서 혼합된 데이터 유형에 대한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Source	문자열 유형	태그	ASCII 값을 포함하는 태그	문자열 유형은 • 기본 STRING 데이터 유형 • 사용자가 작성한 모든 새 문자열 유형
Destination	SINT INT DINT	태그	정수 값을 저장할 태그	Source 값이 부동 소수점 수인 경우 명령어는 대상 데이터 유형에 관계없이 숫자의 비소수 부분만 변환합니다.

식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트) 구문*을 참조하십시오.

설명

STOD 명령어는 Source 를 정수로 변환하고 Destination 에 결과를 저장합니다.

- 이 명령어는 양수 및 음수를 변환합니다.
- Source 문자열에 숫자가 아닌 문자가 포함되어 있으면 STOD 는 첫 번째 연속 숫자 집합을 변환합니다.

이 명령어는 숫자 앞에 있는 빼기 기호를 제외한 최초 제어 문자 또는 숫자가 아닌 문자는 건너뜁니다.

문자열에 구분 기호(예: /)로 구분된 여러 개의 숫자 그룹이 포함되어 있는 경우 명령어는 첫 번째 숫자 그룹만 변환합니다

연산 상태 플래그에 영향

래더 다이어그램에서만. *연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

유형	코드	원인	복구 방법
4	51	문자열 태그의 LEN 값이 문자열 태그의 DATA 크기보다 큽니다.	문자열 유형 태그의 LEN 구성원에 쓰고 있는 명령어가 없는지 확인합니다. LEN 값에 문자열에 포함되는 문자의 수를 입력합니다.
4	53	출력 수가 대상 데이터 유형의 제한을 초과합니다.	• ASCII 값의 크기를 줄이거나 • 대상에 대해 더 큰 데이터 유형을 사용합니다.

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다. Destination 이 지워집니다. 명령어에서 Source 를 변환합니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건	동작
사전 스캔	이전 래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	이전 래더 다이어그램 표의 렁-입력-조건이 참을 참조하십시오.
사후 스캔	이전 래더 다이어그램 표의 사후 스캔을 참조하십시오.

예

MV_read.EM 이 설정된 경우 STOD 명령어는 MV_msg 의 첫 번째 숫자 집합을 정수 값으로 변환합니다. 이 명령어는 최초 제어 문자(\$06)를 건너뛰고 구분 기호(\)에서 중지합니다.

래더 다이어그램

ST(스트럭쳐드 텍스트)

```
IF MV_read.EM THEN
  STOD(MV_msg,MV_msg_nmbr);
  MV_read.EM := 0;
END_IF;
```

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

연산 상태 플래그 페이지의 963

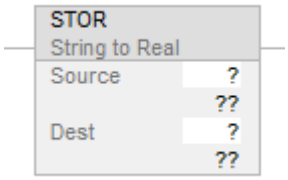
문자열 -> REAL(STOR)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

STOR 명령어는 부동 소수점 값의 ASCII 표현을 REAL 값으로 변환합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

```
STOR(Source,Dest);
```

피연산자

명령어 내에서 혼합된 데이터 유형에 대한 데이터 변환 규칙이 있습니다. *데이터 변환*을 참조하십시오.

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명	참고:
Source	문자열 유형	태그	ASCII 값을 포함하는 태그	문자열 유형은 • 기본 STRING 데이터 유형 • 사용자가 작성한 모든 새 문자열 유형
Destination	REAL	태그	REAL 값을 저장할 태그	

ST(스트럭처드 텍스트) 내에서 식의 구문에 대한 자세한 내용은 ST(스트럭처드 텍스트)를 참조하십시오.

설명

STOR 명령어는 Source 를 REAL 값으로 변환하고 Destination 에 결과를 저장합니다.

- 이 명령어는 양수 및 음수를 변환합니다.
- Source 문자열에 숫자가 아닌 문자가 포함되어 있으면 STOR 은 소수점 [.]을 포함한 첫 번째 연속 숫자 집합을 변환합니다.

이 명령어는 숫자 앞에 있는 빼기 기호를 제외한 최초 제어 문자 또는 숫자가 아닌 문자는 건너뜁니다.

문자열에 구분 기호(예: /)로 구분된 여러 개의 숫자 그룹이 포함되어 있는 경우 명령어는 첫 번째 숫자 그룹만 변환합니다

연산 상태 플래그에 영향

프로그래밍 언어에 따라 조건부. *연산 상태 플래그*를 참조하십시오.

메이저/마이너 폴트

유형	코드	원인	복구 방법
4	51	문자열 태그의 LEN 값이 문자열 태그의 DATA 크기보다 큽니다.	문자열 유형 태그의 LEN 구성원에 쓰고 있는 명령어가 없는지 확인합니다. LEN 값에 문자열에 포함되는 문자의 수를 입력합니다.
4	53	출력 수가 대상 데이터 유형의 제한을 초과합니다.	• ASCII 값의 크기를 줄이거나 • 대상에 대해 더 큰 데이터 유형을 사용합니다.

피연산자 관련 폴트에 대해서는 *공통 속성*을 참조하십시오.

실행

래더 다이어그램

조건	래더 다이어그램 동작
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건	동작
사전 스캔	이전 래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	이전 래더 다이어그램 표의 렁-입력-조건이 참을 참조하십시오.
사후 스캔	이전 래더 다이어그램 표의 사후 스캔을 참조하십시오.

예

저울에서 중량을 읽은 후(weight_read 가 설정 됨) STOR 명령어는 weight_ascii 의 숫자 문자를 REAL 값으로 변환합니다.

Source 와 Destination 의 소수 부분간에 약간의 차이가 있을 수 있습니다.

래더 다이어그램

ST(스트럭쳐드 텍스트)

```
IF weight_read THEN
  STOR(weight_ascii,weight);
END_IF;
```

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

데이터 변환 페이지의 967

연산 상태 플래그 페이지의 963

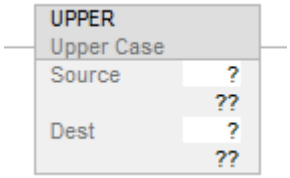
대문자(UPPER)

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다.

UPPER 명령어는 문자열의 알파벳 문자를 대문자로 변환합니다.

사용 가능한 언어

래더 다이어그램



평션 블록

이 명령어는 평션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

UPPER(Source,Dest);

피연산자

래더 다이어그램과 ST(스트럭처드 텍스트)

피연산자	유형	형식	설명
Source	문자열	태그	대문자로 변환할 문자를 포함하는 태그
Destination	문자열	태그	대문자로 문자를 저장하는 태그

ST(스트럭처드 텍스트) 내에서 식의 구문에 대한 자세한 내용은 *ST(스트럭처드 텍스트)*를 참조하십시오.

설명

UPPER 명령어는 Source 의 모든 문자를 대문자로 변환하고 Destination 에 결과를 저장합니다.

- ASCII 문자는 대소문자가 구분됩니다. 대문자 A($41)는 소문자 a($61)와 동일하지 않습니다.
- 작업자가 ASCII 문자를 직접 입력할 경우 비교하기 전에 문자를 모두 대문자 또는 소문자로 변환하십시오.

Source 문자열에서 문자가 아닌 문자는 변경되지 않습니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

유형	코드	원인	복구 방법
4	51	문자열 태그의 LEN 값이 문자열 태그의 DATA 크기보다 큽니다.	문자열 유형 태그의 LEN 구성원에 쓰고 있는 명령어가 없는지 확인합니다. LEN 값에 문자열에 포함되는 문자의 수를 입력합니다.
4	52	출력 문자열이 대상보다 큽니다.	출력 문자열에 대한 충분한 크기의 새 문자열 유형을 만듭니다. 새 문자열 유형을 대상의 데이터 유형으로 사용합니다.

실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A
렁-입력-조건이 참임	명령어가 실행됩니다.
사후 스캔	N/A

ST(스트럭처드 텍스트)

조건	동작
사전 스캔	이전 래더 다이어그램 표의 사전 스캔을 참조하십시오.
정상 실행	이전 래더 다이어그램 표의 렁-입력-조건이 참을 참조하십시오.
사후 스캔	이전 래더 다이어그램 표의 사후 스캔을 참조하십시오.

예

특정 항목에 대한 정보를 찾기 위해 작업자가 항목의 카탈로그 번호를 ASCII 터미널에 입력합니다. 컨트롤러가 터미널에서 입력을 읽은 후에(terminal_read 가 설정 됨), UPPER 명령어는 catalog_number 의 문자를 모두 대문자로 변환하고 결과를 catalog_number_upper_case 에 저장합니다. 그런 다음 후속 렁은

배열에서 catalog_number_upper_case 의 문자와 일치하는 문자를 검색합니다.

래더 다이어그램

ST(스트럭쳐드 텍스트)

```
IF terminal_read THEN
  UPPER(catalog_number,catalog_number_upper_case);
  terminal_read := 0;
END_IF;
```

추가 참조

공통 속성 페이지의 963

ST(스트럭쳐드 텍스트) 구문 페이지의 997

디버그 명령어

디버그 명령어

이 명령어는 개인용 컴퓨터에서 LOGIX 5000 컨트롤러를 에뮬레이션할 수있는 Studio 5000 Logix Emulate 소프트웨어와만 호환됩니다.

로직 상태가 사용자가 결정한 조건에 있을 때 디버그 명령어를 사용하여 이를 모니터링합니다.

사용 가능한 명령어

BPT	TPT

평션 블록

사용할 수 없음

ST(스트럭처드 텍스트)

사용할 수 없음

실행할 작업:	사용할 명령어:
렁이 참일 때 프로그램 에뮬레이이션을 중지합니다.	BPT
렁이 참일 때 선택한 데이터를 기록합니다.	TPT

추가 참조

계산/연산 명령어 페이지의 411

비교 명령어 페이지의 329

비트 명령어 페이지의 81

ASCII 문자열 명령어 페이지의 907

ASCII 변환 명령어 페이지의 927

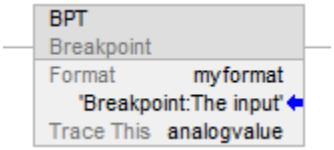
중단점(BPT)

이 명령어는 Studio 5000 Logix Emulate 컨트롤러와만 호환됩니다.

로직 상태가 사용자가 결정한 조건에 있을 때 디버그 명령어를 사용하여 이를 모니터링합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

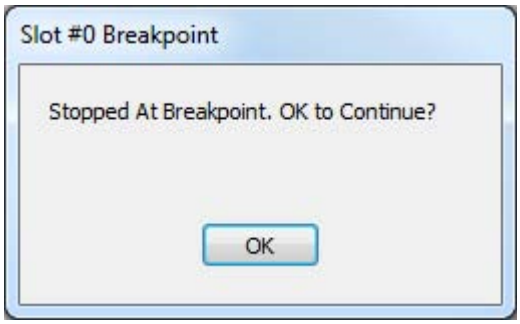
명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Format	문자열	태그	중단점의 추적 창에 나타나는 텍스트의 형식을 설정하는 문자열.
Trace This	BOOL, SINT, INT, DINT, REAL	태그	추적 창에 표시할 값이 있는 태그.

설명

중단점은 중단점 출력 명령어(BPT)로 프로그래밍됩니다. BPT 명령어가 포함된 렁의 입력이 참이면 BPT 명령어는 프로그램 실행을 중지합니다. 소프트웨어는 중단점이 트리거되었음을 나타내는 창과 중단점을 트리거한 값을 표시합니다.



중단점이 트리거되면 에뮬레이터는 중단점이 발생했음을 알리는 창을 표시합니다. 창의 제목 표시줄에는 중단점이 발생한 에뮬레이터가 포함된 슬롯이 표시됩니다.

확인(OK)을 클릭하면 에뮬레이터가 프로그램 실행을 다시 시작합니다. 중단점을 트리거한 조건이 지속되면 중단점이 반복됩니다.

또한 에뮬레이터는 중단점에 대한 추적 창을 엽니다. 추적 창에 중단점 및 값에 대한 정보가 표시됩니다.

중요:	중단점이 트리거되면 실행 계속을 허용할 때까지 프로젝트를 편집할 수 없습니다. 에뮬레이터를 사용하여 온라인 상태로 프로젝트 상태를 관찰할 수 있지만 편집할 수는 없습니다. 중단점이 트리거되는 동안 렁 편집을 허용하려고 하면 컨트롤러가 올바른 모드가 아님을 알리는 대화 상자가 나타납니다

문자열 형식

추적점 및 중단점 명령어에서 Format 문자열을 사용하면 추적된 태그가 추적점 또는 중단점 창에 나타나는 방식을 제어할 수 있습니다. 문자열의 형식은 다음과 같습니다.

- heading:(text)%(type)

여기서 heading 은 추적점 또는 중단점을 식별하는 텍스트 문자열이고, text 는 태그 (또는 선택한 다른 텍스트)를 설명하는 문자열이며, %(type)는 태그의 형식을 나타냅니다. 추적점 또는 중단점 명령어를 사용하여 추적하는 각 태그에 대해 하나의 유형 표시기가 필요합니다.

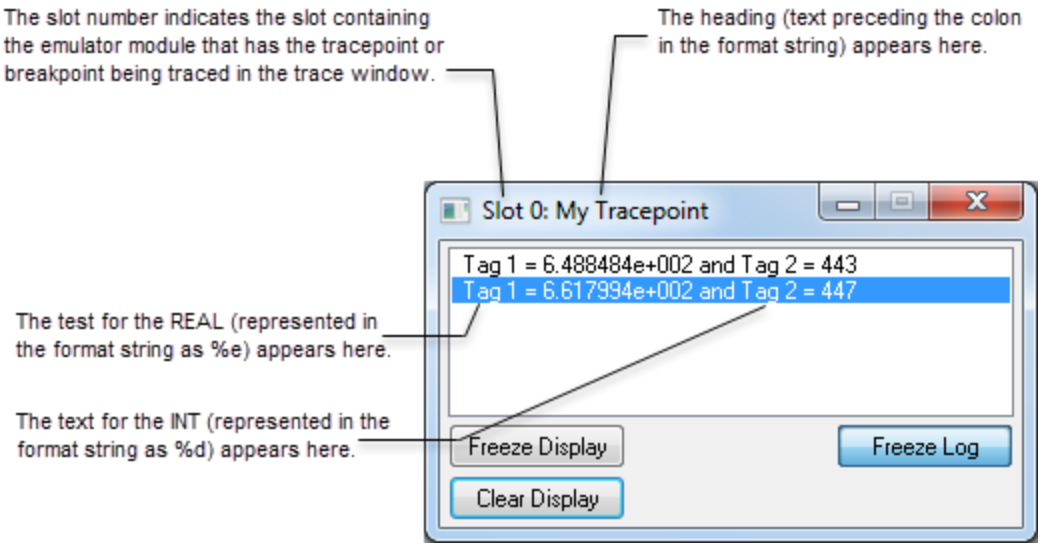
예를 들어, 다음과 같이 추적점 문자열의 형식을 지정할 수 있습니다

- My tracepoint:Tag 1 = %e and Tag 2 = %d

%e 는 첫 번째 추적된 태그를 지수가 있는 배정밀도 부동 소수점 형식으로 지정하고, %d 는 두 번째 추적된 부호를 부호가 있는 십진수 정수로 형식을 지정합니다.

이 경우 두 개의 Trace This 피연산자가 있는 추적점 명령어가 있습니다(모든 태그 값을 임의의 플래그로 형식 지정할 수 있지만, 하나는 REAL 용, 다른 하나는 INT 용).

추적점이 트리거될 때 나타나는 추적점 창은 예와 같습니다.



연산 상태 플래그에 영향

아니요

폴트 조건

이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

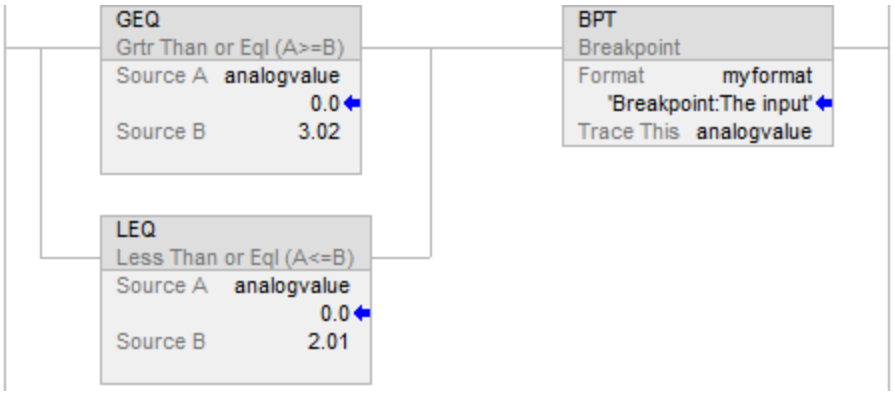
실행

조건	취해진 조치
사전 스캔	렁이 거짓이 됩니다.
렁-입력-조건이 거짓임	렁이 거짓이 됩니다.
렁-입력-조건이 참임	렁이 참이 됩니다. 실행은 참조된 라벨 이름과 함께 LBL 명령어가 포함된 렁으로 이동합니다.
사후 스캔	렁이 거짓이 됩니다.

예

BPT 명령어를 사용하여 많은 태그 값을 표시할 수 있습니다. 그러나 형식 지정 문자열에는 82 자만 포함될 수 있습니다. 형식 지정 문자열에는 중단점에 넣을 각 태그마다 두 개의 문자가 필요하기 때문에 단일 BPT 명령으로 41 개 이상의 태그를 추적할 수 없습니다. 그러나 추적에서 태그 데이터를 구분하려면 공백 및 다른 형식 지정을 포함해야 하므로 하나의 BPT 명령어가 효과적으로 표시할 수 있는 태그 값 수를 41 개보다 훨씬 적게 줄입니다.

이 렁은 아날로그 값이 3.02 보다 크거나 2.01 보다 작으면 프로그램 실행을 중지시키는 중단점을 표시합니다.



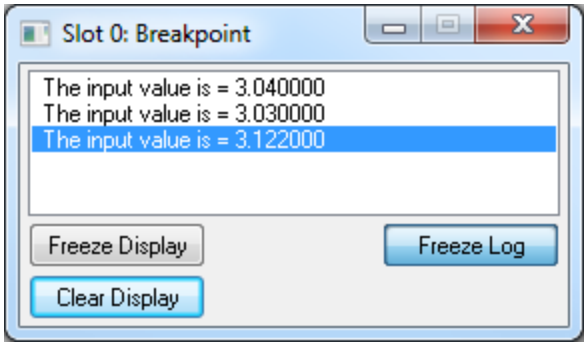
중단점 정보를 형식 문자열(myformat)로 표시하십시오. 이 경우 형식 문자열에 다음 텍스트가 포함됩니다.

- Breakpoint:The input value is %f

중단점이 트리거되면 중단점 추적 창의 제목 표시줄에 콜론 앞에 있는 문자('Breakpoint')가 표시됩니다. 다른 문자가 추적을

구성합니다. 이 예에서 %f 는 추적할 첫 번째 태그(이 경우 유일한 태그)를 나타냅니다('analogvalue').

결과 추적은 다음과 같이 표시됩니다.



추가 참조

공통 속성 페이지의 963

데이터 변환 페이지의 967

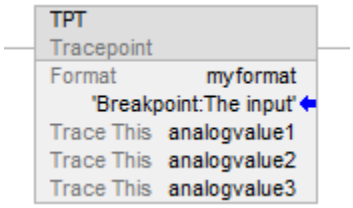
추적점(TPT)

이 명령어는 Studio 5000 Logix Emulate 컨트롤러와만 호환됩니다.

추적점 명령어는 렁이 참일 때 선택한 데이터를 기록합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭처드 텍스트)

이 명령어는 ST(스트럭처드 텍스트)에서 사용할 수 없습니다.

피연산자

명령어 내에서 혼합된 데이터 유형을 위한 데이터 변환 규칙이 있습니다. 데이터 변환을 참조하십시오.

래더 다이어그램

피연산자	유형	형식	설명
Format	문자열	태그	추적 보고서의 형식을 설정하는 문자열(화면에 표시되고 디스크에 기록됨).
Trace This	BOOL SINT INT DINT REAL	태그	추적할 태그.

설명

추적점은 추적점 출력 명령어(TPT)으로 프로그래밍됩니다. TPT 명령어가 포함된 렁에서 입력이 참이면 TPT 명령어는 추적 항목을 추적 디스플레이 또는 로그 파일에 씁니다.

TPT 명령어를 사용하여 많은 태그를 추적할 수 있습니다. 그러나 형식 지정 문자열에는 82 자만 포함될 수 있습니다. 형식 지정 문자열에는 추적하려는 각 태그마다 두 개의 문자가 필요하기 때문에 단일 TPT 명령으로 41 개 이상의 태그를 추적할 수 없습니다. 그러나 추적에서 태그 데이터를 구분하려면 공백 및 다른 형식 지정을 포함해야 하므로 하나의 TPT 명령어가 효과적으로 추적할 수 있는 태그 수를 41 개보다 훨씬 적게 줄입니다.

문자열 형식

추적점 및 중단점 명령어에서 Format 문자열을 사용하면 추적된 태그가 추적점 또는 중단점 창에 나타나는 방식을 제어할 수 있습니다. 문자열의 형식은 다음과 같습니다.

- heading:(text)%(type)

여기서 heading 은 추적점 또는 중단점을 식별하는 텍스트 문자열이고, text 는 태그 (또는 선택한 다른 텍스트)를 설명하는 문자열이며, %(type)는 태그의 형식을 나타냅니다. 추적점 또는 중단점 명령어를 사용하여 추적하는 각 태그에 대해 하나의 유형 표시기가 필요합니다.

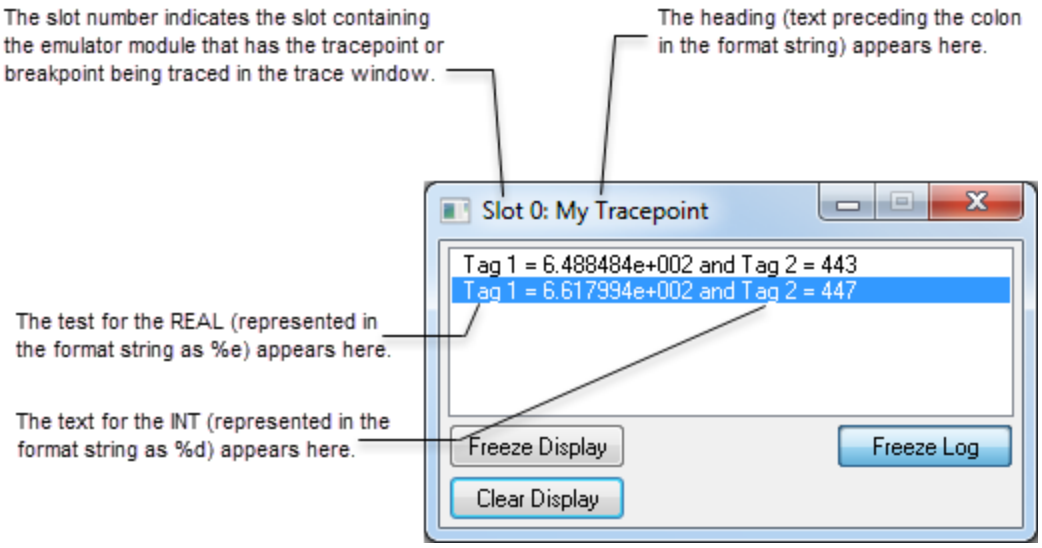
예를 들어, 다음과 같이 추적점 문자열의 형식을 지정할 수 있습니다

- My tracepoint:Tag 1 = %e and Tag 2 = %d

%e 는 첫 번째 추적된 태그를 지수가 있는 배정밀도 부동 소수점 형식으로 지정하고, %d 는 두 번째 추적된 부호를 부호가 있는 십진수 정수로 형식을 지정합니다.

이 경우 두 개의 Trace This 피연산자가 있는 추적점 명령어가 있습니다(모든 태그 값을 임의의 플래그로 형식 지정할 수 있지만, 하나는 REAL 용, 다른 하나는 INT 용).

추적점이 트리거될 때 나타나는 추적점 창은 예와 같습니다.



연산 상태 플래그에 영향

아니요

폴트 조건

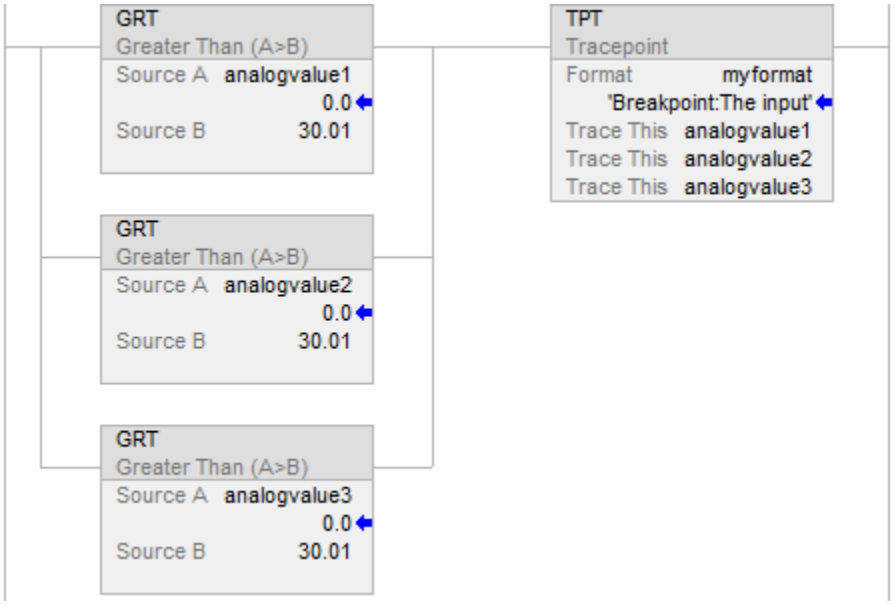
이 명령어에만 해당되는 것은 없습니다. 피연산자 관련 폴트에 대해서는 공통 속성을 참조하십시오.

실행

조건	릴레이 래더 동작
사전 스캔	렁이 거짓이 됩니다.
렁-입력-조건이 거짓임	렁이 거짓이 됩니다.
렁-입력-조건이 참임	렁이 참이 됩니다. 실행은 참조된 라벨 이름과 함께 LBL 명령어가 포함된 렁으로 이동합니다.
사후 스캔	렁이 거짓이 됩니다.

예

이 렁은 그 중 하나가 주어진 값(30.01)을 초과할 때 세 가지 아날로그 값의 추적을 트리거합니다.



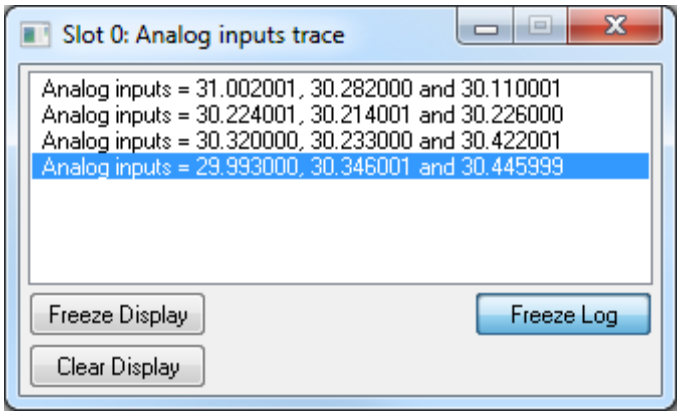
추적점 정보를 형식 문자열(myformat)로 표시합니다.

이 경우 형식 문자열에 다음 텍스트가 포함됩니다.

- Analog inputs trace:Analog inputs = %f, %f, and %f

추적점이 트리거되면 콜론 앞의 문자('Analog inputs trace')가 추적 창의 제목 표시 줄에 나타납니다. 다른 문자가 추적을 구성합니다. 이 예에서 %f 는 추적할 태그를 나타냅니다('analogvalue1', 'analogvalue2' 및 'analogvalue3').

결과 추적은 다음과 같이 표시됩니다.



이 추적이 디스크에 기록되면 콜론 앞에 있는 문자가 추적에 표시됩니다.

이는 어느 추적점이 어느 추적 항목을 가져왔는지 나타냅니다. 이것이 추적 항목의 예입니다. 'Analog inputs trace:'는 추적점 형식 문자열의 제목 텍스트입니다.

Analog inputs trace: Analog inputs = 31.00201, 30.282000, and 30.110001.

추가 참조

디버그 명령어 페이지의 949

중단점(BPT) 페이지의 950

공통 속성 페이지의 963

데이터 변환 페이지의 967

사용권 명령어

사용권 명령어는 프로젝트에 사용된 사용권을 확인하는 데 사용됩니다.

사용 가능한 언어

래더 다이어그램

LV

펑션 블록

사용할 수 없음

ST(스트럭처드 텍스트)

사용할 수 없음

추가 참조

연산 변환 명령어 페이지의 837

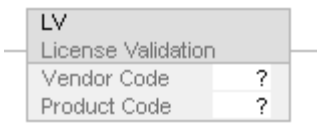
사용권 유효성 검사(LV)

이 정보는 Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580 및 GuardLogix 5580 컨트롤러에 적용됩니다.

라이센스 유효성 검사(LV) 명령어는 루틴 또는 애드온 명령어와 연결된 만료되지 않은 라이센스가 컨트롤러에 있는지 확인합니다.

사용 가능한 언어

래더 다이어그램



펑션 블록

이 명령어는 펑션 블록에서 사용할 수 없습니다.

ST(스트럭쳐드 텍스트)

이 명령어는 ST(스트럭쳐드 텍스트)에서 사용할 수 없습니다.

피연산자

래더 다이어그램

피연산자	유형	형식	설명
공급업체 코드(Vendor Code)	DINT	즉시	루틴 또는 애드온 명령어와 연결된 라이센스의 공급업체를 식별하는 고유 번호입니다. 0 ~ 2,147,483,647 범위의 즉시 정수 값이 허용됩니다.
제품 코드(Product Code)	DINT	즉시	루틴 또는 애드온 명령어와 연결된 라이센스의 제품 코드를 식별하는 고유 번호입니다. 0 ~ 2,147,483,647 범위의 즉시 정수 값이 허용됩니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

이 명령어에만 해당되는 것은 없습니다.

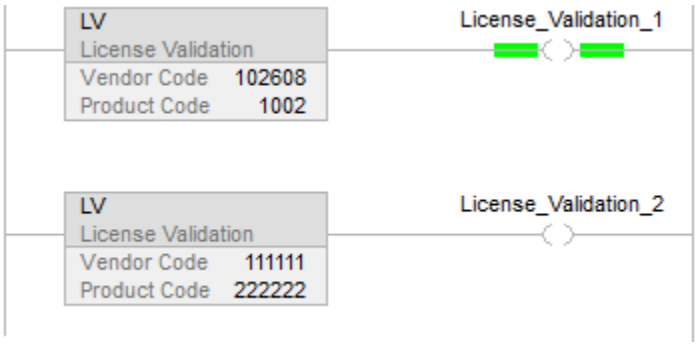
실행

래더 다이어그램

조건/상태	취해진 조치
사전 스캔	N/A
렁-입력-조건이 거짓임	N/A

조건/상태	취해진 조치
렁-입력-조건이 참임	숫자 비교: 라이센스가 유효하고 프로젝트에서 사용된 경우 렁-출력-조건을 참으로 설정 else 렁-출력-조건을 거짓으로 해제
사후 스캔	N/A

예



추가 참조

사용권 명령어 페이지의 959

일반 명령어의 공통 특성

일반 명령어의 공통 특성은 본 장의 가이드라인을 따르십시오.

공통 특성

LOGIX 5000™ 명령어에 일반적인 속성에 대한 자세한 내용을 확인하려면 아래 항목을 클릭하십시오.

연산 상태 플래그 페이지의 963

즉시 값 페이지의 967

데이터 변환 페이지의 967

기본 데이터 유형 페이지의 972

LINT 데이터 유형 페이지의 975

부동 소수점 값 페이지의 976

배열을 통한 인덱스 페이지의 978

비트 주소 지정 페이지의 980

연산 상태 플래그

연산 상태 플래그에 대해 이 항목에 나와 있는 지침을 따르십시오.

설명

컨트롤러	설명
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	명령어로 직접 액세스를 위한 연산 상태 플래그의 집합 이 플래그는 래더 다이어그램 루틴에서만 업데이트되며 태그가 아닙니다. 플래그 별칭은 적용되지 않습니다.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	명령어로 직접 액세스를 위한 연산 상태 플래그의 집합 이 플래그는 모든 루틴 유형에서 업데이트되지만 태그가 아닙니다. 플래그 별칭은 적용되지 않습니다.

상태 플래그

상태 플래그	설명 (CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러)	설명 (CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러)
S:FS 최초 스캔 플래그	다음과 같은 경우 최초 스캔 플래그는 컨트롤러에 의해 설정됩니다. • 컨트롤러가 실행 모드로 전환된 후 처음으로 프로그램이 스캔되는 경우 • 컨트롤러가 금지 해제된 후 처음으로 프로그램이 스캔되는 경우 • 루틴이 SFC 작업에서 호출되고 이 작업을 소유한 단계가 처음으로 스캔되는 경우 첫 스캔 플래그를 사용하여 이후 스캔에 사용하도록 데이터를 초기화합니다. 이를 최초 통과 비트라고도 합니다.	다음과 같은 경우 최초 스캔 플래그는 컨트롤러에 의해 설정됩니다. • 컨트롤러가 실행 모드로 전환된 후 처음으로 프로그램이 스캔되는 경우 • 컨트롤러가 금지 해제된 후 처음으로 프로그램이 스캔되는 경우 • 루틴이 SFC 작업에서 호출되고 이 작업을 소유한 단계가 처음으로 스캔되는 경우 이 플래그를 사용하여 이후 스캔에 사용하도록 데이터를 초기화합니다. 이를 최초 통과 비트라고도 합니다.
S:N 음의 플래그	컨트롤러에서 산술 연산 또는 로직 작업의 결과가 음의 값이면 음의 플래그를 설정합니다. 이 플래그를 사용하여 음의 값을 신속하게 테스트해볼 수 있습니다.	컨트롤러에서 산술 연산 또는 로직 작업의 결과가 음의 값이면 음의 플래그를 설정합니다. 이 플래그를 사용하여 음의 값을 신속하게 테스트해볼 수 있습니다. S:N 을 사용하면 CMP 명령어를 사용할 때보다 효과적입니다.
S:Z 0 플래그	0 플래그는 산술 연산 또는 로직 작업의 결과가 0 일 때 컨트롤러에서 설정됩니다. 이 플래그를 사용하여 0 값을 신속하게 테스트해볼 수 있습니다. 0 플래그는 플래그를 설정할 수 있는 명령어가 실행되자마자 해제됩니다.	컨트롤러에서 수학 또는 로직 연산의 결과가 0 이면 0 플래그를 설정합니다. 이 플래그를 사용하여 0 값을 신속하게 테스트해볼 수 있습니다.

S:V 오버플로 플래그	오버플로 플래그는 다음과 같은 경우 컨트롤러에서 설정합니다. • 수학 연산의 결과는 오버플로입니다. 예를 들어, SINT 에 1 을 더했을 때 값이 127 ~ -128 이면 오버플로가 생성됩니다. • 대상 태그가 너무 작아 값을 보유하지 못하는 경우. 예를 들어 SINT 또는 INT 태그에 값 123456 을 저장하려는 경우가 해당됩니다. 이 오버플로 플래그를 사용하여 연산 결과가 범위 내에 있는지 확인합니다. 저장 중인 데이터가 문자열 형식일 때, 그 문자열이 너무 커 대상 태그에 맞지 않는다면 S:V 가 설정됩니다. **팁:** 해당한다면, OTE 또는 OTL 명령어로 S:V 를 설정하십시오. 오버플로 폴트 보고를 활성화 또는 비활성화하려면 **컨트롤러 속성 > 고급 탭 > 오버플로 폴트 보고**를 클릭하십시오. 배열 첨자를 평가하는 동안 오버플로가 발생하면 마이너 폴트가 생성되고 메이저 폴트가 생성돼 인덱스가 범위를 벗어났음을 알립니다.	오버플로 플래그는 다음과 같은 경우 컨트롤러에서 설정합니다. • 산술 연산의 결과로 오버플로가 발생하는 경우. 예를 들어, 값이 -128 ~ 127 일 때 SINT 에 1 을 더하면 오버플로가 생성됩니다. • 대상 태그가 너무 작아 값을 보유하지 못하는 경우. 예를 들어 SINT 또는 INT 태그에 값 123456 을 저장하려는 경우가 해당됩니다. 이 오버플로 플래그를 사용하여 연산 결과가 범위 내에 있는지 확인합니다. 오버플로 플래그가 설정될 때마다 마이너 폴트가 발생합니다. **팁:** 해당한다면, OTE 또는 OTL 명령어로 S:V 를 설정하십시오.
S:C 자리올림 플래그	산술 연산의 결과로 최상위 비트의 자리올림이 발생한 경우 컨트롤러에서 자리올림 플래그를 설정합니다. 정수 값으로 된 ADD 및 SUB 명령어만 이 플래그에 영향을 줍니다. +와 - 연산자는 그렇지 않습니다.	산술 연산의 결과로 최상위 비트의 자리올림이 발생한 경우 컨트롤러에서 자리올림 플래그를 설정합니다.

S:MINOR 마이너 폴트 플래그	최소 하나의 부 프로그램 폴트가 있을 때 컨트롤러에서 마이너 폴트 플래그를 설정합니다. 마이너 폴트 태그로 어떤 마이너 폴트의 발생 여부를 검증합니다. 이 비트는 오버플로와 같은 프로그래밍 폴트로만 트리거됩니다. 배터리 폴트로는 트리거되지 않습니다. 스캔이 시작할 때마다 이 비트는 해제됩니다. **팁:** 해당한다면, OTE 또는 OTL 명령어로 분명하게 S:MINOR 를 설정하십시오.	최소 하나의 부 프로그램 폴트가 있을 때 컨트롤러에서 마이너 폴트 플래그를 설정합니다. 마이너 폴트 플래그로 마이너 폴트 발생 여부를 테스트하고 적절한 조치를 취하십시오. 이 비트는 오버플로 같은 프로그래밍 언어로만 트리거됩니다. 배터리 폴트로는 트리거되지 않습니다. 스캔이 시작할 때마다 이 비트는 해제됩니다. **팁:** 해당한다면, OTE 또는 OTL 명령어로 분명하게 S:MINOR 를 설정하십시오.
중요:	연산 상태 플래그는 저장된 값에 따라 설정됩니다. 유형 변환이 혼합 데이터 유형에서 명령어 파라미터에 대해 발생하면 평상시 연산 상태 플래그에 영향을 주지 않는 명령어가 연산 상태 플래그에 영향을 줄 수 있습니다. 유형 변환 프로세스에서 연산 상태 플래그를 설정합니다.	

배열 첨자의 식

컨트롤러	**설명**
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러	식이 산술 연산 결과에 따라 상태 플래그를 설정하지 않습니다. 식이 오버플로되는 경우: • 컨트롤러가 마이너 폴트를 생성하도록 구성되면 마이너 폴트가 생성됩니다. • 결과 값이 범위를 벗어나기 때문에 메이저 폴트(유형 4, 코드 20)가 생성됩니다.
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	식이 산술 연산 결과에 따라 상태 플래그를 설정합니다. 배열 첨자가 식이면, 그 식과 명령어가 마이너 폴트를 생성할 수도 있습니다.

팁: 배열 첨자가 너무 크면(범위 이탈), 메이저 폴트(유형 4, 코드 20)가 생성됩니다.

즉시 값

즉시 값(상수)을 십진수 형식(예: -2, 3)으로 입력하면 컨트롤러에서는 32 비트를 사용하여 값을 저장합니다. 값은 십진수가 아닌 기수(예: 이진수, 16 진수)로 입력하고 32 비트 중 일부를 지정하지 않으면 컨트롤러에서는 지정하지 않은 비트에 0 을 배치합니다(영 채우기).

중요: 32 비트보다 작은 즉시 2 진, 8 진, 16 진 값의 영 채우기

다음 값을 입력하는 경우	컨트롤러에서 저장하는 값
-1	16#ffff ffff (-1)
16#ffff (-1)	16#0000 ffff (65535)
8#1234 (668)	16#0000 029c (668)
2#1010 (10)	16#0000 000a (10)

정수 즉시 값

다음 값을 입력하는 경우	컨트롤러에서 저장하는 값
접미사가 없는 값	DINT
"U"	UDINT
"L"	LINT
"UL"	ULINT

부동 소수점 즉시 값

다음 값을 입력하는 경우	컨트롤러에서 저장하는 값
접미사가 없는 값	REAL
"L"	LREAL

데이터 변환

프로그래밍에서 데이터 유형을 혼합하면 데이터 변환이 발생합니다.

프로그래밍 대상	변환이 발생하는 경우
래더 다이어그램 ST(스트럭쳐드 텍스트)	하나의 명령어 내에서 파라미터의 데이터 유형을 명령어 또는 식.
펑션 블록	데이터 유형이 다른 파라미터 2 개를 연결하는 경우

명령어의 모든 피연산자에서 다음과 같은 데이터 유형을 사용하는 경우 명령어 실행 속도가 빨라지고 필요한 메모리가 줄어듭니다.

- 동일한 데이터 유형
- 중간 데이터 유형:
  - 모든 평션 블록 명령어는 하나의 데이터 유형 피연산자만 지원합니다.
  - 데이터 유형을 혼합하거나 최적의 데이터 유형이 아닌 태그를 사용하면 컨트롤러에서는 다음 규칙에 따라 데이터를 변환합니다.
    - 피연산자는 SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL 의 데이터 유형 순위에 따라 변환되며 데이터 유형의 순위는 1(가장 낮음)부터 10(가장 높음)까지 있습니다.

      팁: 데이터 변환에 필요한 시간과 메모리를 줄이려면 명령어의 모든 피연산자에 대해 동일한 데이터 유형을 사용하십시오.

SINT 또는 INT 를 DINT 로 변환하거나 DINT 를 LINT 로 변환

SINT 또는 INT 입력 소스 태그는 소스 태그에 대한 부호 확장을 통해 DINT 값으로 승격됩니다. SINT 또는 INT 값을 DINT 값으로 변환하는 명령어는 다음 변환 방법 중 하나를 사용합니다.

변환 방법	데이터 변환을 위한 배치 방법
부호 확장	32 비트 또는 64 비트가 될 때까지 기존 비트 왼쪽에 있는 각 비트 위치로 맨 왼쪽 비트(값의 기호)의 값 배치.
영 채우기	32 비트 또는 64 비트가 될 때까지 기존 비트 왼쪽에 0 배치.

로직 명령어는 영 채우기를 사용합니다. 기타 모든 명령어는 부호 확장을 사용합니다.

다음은 부호 확장 및 영 채우기를 사용하여 값을 변환한 결과를 보여주는 예입니다.

값	2#1111_1111_1111_1111	(-1)
부호 확장을 통해 이 값으로 변환	2#1111_1111_1111_1111_1111_1111_1111_1111	(-1)
영 채우기를 통해 이 값으로 변환	2#0000_0000_0000_0000_1111_1111_1111_1111	(65535)

부호 확장을 통해 데이터를 변환하는 명령어에서 SINT 또는 INT 태그와 즉시 값을 사용하는 경우 다음 방법 중 하나를 사용하여 즉시 값을 처리하십시오.

즉시 값을 십진 기수로 지정합니다.

값을 십진수가 아닌 기수로 입력하는 경우 즉시 값의 32 비트를 모두 지정합니다. 이렇게 하려면 32 비트가 될 때까지 값의 왼쪽에 있는 각 비트 위치에 맨 왼쪽 비트의 값을 입력합니다.

각 연산자에 대한 태그를 생성하고 명령어 전체에서 동일한 데이터 유형을 사용합니다. 상수 값을 할당하려면 다음 중 하나를 수행하십시오.

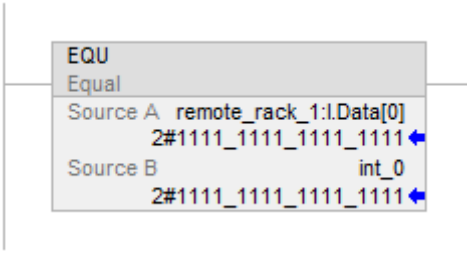
태그 중 하나에 상수 값을 입력합니다.

태그 중 하나로 값을 이동하는 MOV 명령어를 추가합니다.

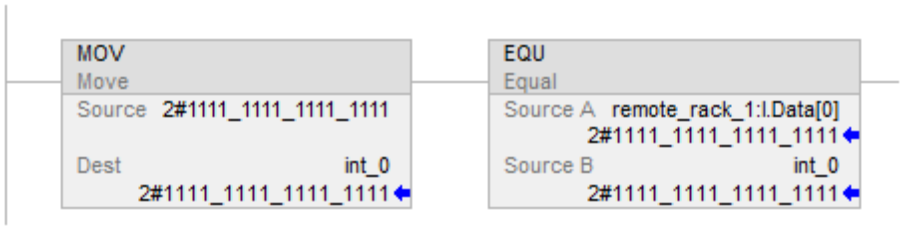
MEQ 명령어를 사용하여 필요한 비트만 확인합니다.

다음은 즉시 값을 INT 태그와 혼합하는 두 가지 방법을 보여주는 예입니다. 두 가지 예에서는 1771 I/O 모듈의 비트를 검사하여 모든 비트가 제자리에 있는지 확인합니다. 1771 I/O 모듈의 입력 데이터 워드가 INT 태그이기 때문에 16 비트 상수 값을 사용하는 것이 가장 쉽습니다.

중요:	즉시 값과 INT 태그 혼합 remote_rack_1:I.Data[0]가 INT 태그이므로 이를 검사하는 값이 역시 INT 태그로 입력됩니다.



중요:	즉시 값과 INT 태그 혼합 remote_rack_1:I.Data[0]가 INT 태그이므로 이를 검사하는 값이 먼저 int_0 으로 이동하고 이는 역시 INT 태그입니다. 그런 다음 EQU 명령어가 두 태그를 비교합니다.



REAL 로 정수 변환

컨트롤러에서는 REAL 값을 IEEE 단정도 부동 소수점 수 형식으로 저장합니다. 값의 부호에 1 비트를, 기준 값에 23 비트를, 지수에 8 비트를 사용합니다(총 32 비트). 동일한 명령어에 정수 태그(SINT, INT 또는 DINT)와 REAL 태그를 혼합하여 입력하면 컨트롤러에서는 명령어를 실행하기 전에 정수 값을 REAL 값으로 변환합니다.

- SINT 또는 INT 값은 항상 동일한 REAL 값으로 변환됩니다.
- DINT 값은 동일한 REAL 값으로 변환되지 않을 수 있습니다.
- REAL 값은 기준 값에 최대 24 비트를 사용합니다(저장된 23 비트 + '숨겨진' 1 비트).
- DINT 값은 값에 최대 32 비트를 사용합니다(부호에 1 비트, 값에 31 비트).

DINT 값에 25 개 이상의 유효 비트가 필요한 경우 이 값은 동일한 REAL 값으로 변환되지 않을 수 있습니다. 동일한 REAL 값으로 변환되지 않으면 컨트롤러에서는 가장 근사한 짝수 값으로 반올림된 가장 높은 24 비트를 저장합니다.

DINT 를 SINT 또는 INT 로 변환

DINT 값을 SINT 또는 INT 값으로 변환하기 위해 컨트롤러에서는 DINT 의 위쪽 부분을 잘라내고 데이터 유형에 맞는 낮은 비트를 저장합니다. 값이 너무 크면 변환 시 오버플로가 발생합니다.

	DINT 를 INT 및 SINT 로 변환	
DINT 값	다음 더 작은 값으로 변환	
16#0001_0081 (65,665)	INT:	16#0081 (129)
	SINT:	16#81 (-127)

REAL 을 SINT, INT 또는 DINT 로 변환

REAL 값을 정수로 변환하기 위해 컨트롤러에서는 소수 부분을 반올림하여 결과 데이터 유형에 맞는 비트를 저장합니다. 값이 너무 크면 변환 시 오버플로가 발생합니다.

숫자는 다음 예처럼 반올림됩니다.

0.5 보다 작은 소수는 가장 근사한 정수 값으로 내립니다.

0.5 보다 큰 소수는 가장 근사한 정수 값으로 반올립니다.

0.5 는 가장 근사한 짝수 값으로 반올리거나 내립니다.

중요:	REAL 값을 DINT 값으로 변환
REAL 값	**다음 DINT 값으로 변환**
-2.5 -3.5	-2 -4
-1.6	-2
-1.5	-2
-1.4	-1
1.4	1
1.5	2
1.6	2
2.5 3.5	2 4

기본 데이터 유형

컨트롤러는 IEC 1131-3 에 정의된 기본 데이터 유형을 지원합니다. 기본 데이터 유형은 다음과 같습니다.

데이터 유형	설명	범위
BOOL	1 비트 부울	0 = 해제 1 = 설정
SINT	1 바이트 정수	-128 ~ 127
INT	2 바이트 정수	-32,768 ~ 32,767
DINT	4 바이트 정수	-2,147,483,648 ~ 2,147,483,647
REAL	4 바이트 부동 소수점 수	$-3.402823E^{38}$ ~ $-1.1754944E^{-38}$ (음수 값) 및 0 및 $1.1754944E^{-38}$ ~ $3.402823E^{38}$ (양수 값)
LINT	8 바이트 정수	0~32,535,129,599,999,999
USINT	1 바이트 부호 없는 정수	0 ~ 255
UINT	2 바이트 부호 없는 정수	0 ~ 65,535
UDINT	4 바이트 부호 없는 정수	0 ~ 4,294,967,295
ULINT	8 바이트 부호 없는 정수	0 ~ 18,446,744,073,709,551,615
REAL	4 바이트 부동 소수점 수	-3.4028235E38 ~ -1.1754944E-38 (음수 값) 및 0.0 및 1.1754944E-38 ~ 3.4028235E38 (양수 값)
LREAL	8 바이트 부동 소수점 수	-1.7976931348623157E308 ~ -2.2250738585072014E-308 (음수 값) 및 0.0 및 2.2250738585072014E-308 ~ 1.7976931348623157E308 (양수 값)

이러한 컨트롤러는 다음과 같은 기본 데이터 유형을 지원합니다.

컨트롤러	데이터 유형
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	SINT, INT, DINT, LINT, REAL USINT, UINT, UDINT, ULINT, LREAL
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	SINT, INT, DINT, LINT, REAL.

컨트롤러는 모든 즉시 값을 DINT 데이터 유형으로 처리합니다.

REAL 데이터 유형은 ±infinity 및 ±NAN 도 저장하지만 소프트웨어 디스플레이는 디스플레이 형식에 따라 다릅니다.

데이터 유형 변환

하나의 명령어 내에서 피연산자의 데이터 유형을 혼합하는 경우 명령어에 따라 데이터가 해당 명령어에 적합한 최적의 데이터 유형으로 자동 변환됩니다. 컨트롤러가 새 데이터 유형에 맞춰 데이터를 변환하거나 단순히 가능한 최적의 형태로 데이터를 끼워 맞추는 경우도 있습니다.

변환	결과		
큰 정수 -> 작은 정수	컨트롤러는 큰 정수의 위쪽 부분을 잘라서 오버플로를 만듭니다. 예:		
	10 진수		**이진수**
	DINT	65,665	0000_0000_0000_0001_0000_0000_1000_0001
	INT	129	0000_0000_1000_0001
	SINT	-127	1000_0001
SINT 또는 INT -> REAL	데이터 정밀도가 상실되지 않음		
DINT -> REAL	데이터 정밀도가 상실되었을 수 있음. 두 데이터 유형 모두 32 비트 데이터를 저장하지만 REAL 유형은 32 비트 중 일부를 지수 값을 저장하는 데 사용합니다. 정밀도를 상실한 경우 컨트롤러는 DINT 의 최하위 비트를 참조합니다.		
LREAL -> LREAL	데이터 정밀도가 상실되지 않음.		
LREAL -> REAL	데이터 정밀도가 상실되었을 수 있음.		

LREAL/REAL -> 부호 없는 정수	데이터 정밀도가 상실되었을 수 있음. 소스 값이 너무 커서 대상에 맞지 않을 경우 컨트롤러가 저장할 수 있는 부분만 저장하고 오버플로를 생성할 수 있습니다.
부호 있는 정수/부호 없는 정수 -> LREAL/REAL	대상에 저장할 수 있는 것보다 더 많은 상위 비트가 정수 값이 포함된 경우 하위 비트가 잘립니다.
부호 있는 정수 -> 부호 없는 정수	소스 값이 너무 커서 대상에 맞지 않을 경우 컨트롤러가 저장할 수 있는 부분만 저장하고 오버플로를 생성할 수 있습니다.
부호 없는 정수 -> 부호 있는 정수	소스 값이 너무 커서 대상에 맞지 않을 경우 컨트롤러가 저장할 수 있는 부분만 저장하고 오버플로를 생성할 수 있습니다.
정수로 REAL 변환	컨트롤러는 분수 부분을 반올림하고 분수가 아닌 부분의 위쪽 부분을 자릅니다. 데이터 손실 시 컨트롤러는 오버플로 상태 플래그를 설정합니다. 최대한 가까운 정수로 반올림합니다. 0.5 보다 작으면 버리고 0.5 인 경우 우수리 없이 최대한 가까운 정수, 0.5 보다 큰 경우 반올림 예:

REAL(소스)	DINT(결과)
1.6	2
-1.6	-2
1.5	2
-1.5	-2
1.4	1
-1.4	-1
2.5	2
-2.5	-2

데이터를 BOOL 데이터 유형으로 변환하거나 BOOL 데이터 유형에서 변환하는 것은 불가능합니다.

중요: 연산 상태 플래그는 저장되는 값에 따라 설정됩니다. 일반적으로 수학 상태 키워드에 영향을 미치지 않는 명령어이지만 명령어 파라미터의 데이터 유형이 혼합되어 유형 변환이 일어나는 경우 영향을 미치는 것처럼 보일 수 있습니다. 유형 변환 프로세스를 통해 수학 상태 키워드가 설정됩니다.

안전 데이터 유형

Logix Designer 응용 프로그램은 안전 태그가 직간접적으로 참조하는 사용자 정의 또는 애드온(Add-On) 정의 유형에 잘못된 데이터 유형을 포함시킬 수 있는 사용자 정의 또는 애드온 정의

유형의 수정 작업을 금지합니다. (여기에는 중첩된 구조가 포함됩니다.)

안전 태그는 다음과 같은 데이터 유형으로 구성될 수 있습니다.

- 모든 기본 데이터 유형
- 안전 명령어에 사용되는 미리 정의된 유형.
- 사용자 정의 데이터 유형 또는 앞의 두 유형으로 구성된 배열.

안전 태그의 UDT 구성원 이름 온라인 편집

CompactLogix 5380, Compact GuardLogix 5380, CompactLogix 5480, ControlLogix 5580 및 GuardLogix 5580 컨트롤러에서 사용자 정의 데이터 유형의 구성원 이름을 온라인으로 편집할 수 있습니다. 그러나 사용자 정의 데이터 유형이 안전 태그에 사용되고 컨트롤러가 안전 보안 상태인 경우에는 온라인 편집이 비활성화됩니다.

추가 참조

연산 상태 플래그 페이지의 963

LINT 데이터 유형

LINT 데이터 유형은 64 비트 정수입니다.

LINT 데이터 유형은 Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580 또는 GuardLogix 5580 컨트롤러의 여러 명령어에서 사용할 수 있지만 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570 컨트롤러의 대부분의 명령어에서는 사용할 수 없습니다.

CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570 컨트롤러에서 LINT 데이터 유형을 사용할 경우 다음을 고려하십시오.

팁: LINT 는 복사(COP, CPS) 명령어에서만 사용할 수 있습니다. 이러한 명령어는 CST/WallClock 시간 속성, 시간 동기화, Add-On 명령어와 함께 사용됩니다. 이 태그 유형을 더하거나, 빼거나, 곱하거나, 나눌 수 없습니다.

LINT 데이터 유형을 사용하는 경우 아래 문제가 발생하면 다음 설명을 참조하십시오.

방법	설명
이중 정수 DINT 값 2 개를 LINT 하나로 이동/복사	총 64 비트가 되도록(즉, DINT[2]) 2 개의 요소로 구성된 배정도 정수 배열을 만듭니다. 그런 다음 긴 정수 하나로 복사할 수 있습니다.
날짜/시간 표시 에러 수정	태그에 음의 값이 있으면 날짜/시간으로 표시할 수 없습니다. 태그 편집기에서 태그 스타일을 날짜/시간에서 2 진수로 변경하여 이 값이 음수인지 확인합니다. 최상위 비트(맨 왼쪽 비트)가 1 이면 해당 값이 음수이므로 날짜 또는 시간으로 표시할 수 없습니다.

부동 소수점 값

이 정보는 CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370, GuardLogix 5570, Compact GuardLogix 5380, CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, GuardLogix 5580 컨트롤러에 적용됩니다. 컨트롤러 구분이 있을 때는 언급합니다.

Logix 컨트롤러는 부동 소수점 연산을 위한 IEEE 754 표준에 따라 부동 소수점 값을 처리합니다. 이 표준은 부동 소수점 수가 저장 및 계산되는 방법을 정의합니다. 부동 소수점 연산을 위한 IEEE 754 표준은 적절한 저장 공간에서 매우 큰 숫자를 처리할 수 있는 기능 및 속도를 제공하기 위해 마련되었습니다.

REAL 태그는 정규화 단정도 부동 소수점 값을 저장합니다.

LREAL 태그는 정규화 배정도 부동 소수점 값을 저장합니다.

컨트롤러는 다음과 같은 기본 데이터 유형을 지원합니다.

컨트롤러	데이터 유형
CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러	REAL, LREAL
CompactLogix 5370, ControlLogix 5570, Compact GuardLogix 5370 및 GuardLogix 5570 컨트롤러	REAL

비정규화 값 및 -0.0 은 0.0 으로 처리됩니다.

계산 결과가 NAN 값이면 부호 비트가 양의 부호 또는 음의 부호일 수 있습니다. 이러한 경우 소프트웨어에서는 부호 없이 1#.NAN 을 표시합니다.

일부 십진수 값은 표준 형식으로 정확하게 표현할 수 없으며, 이러한 경우 정밀도가 손실됩니다. 예를 들어, 10.1 에서 10 을 빼면 결과는 0.1 입니다. 그러나 Logix 컨트롤러에서 이 결과는 0.10000038 이 되기가 아주 쉽습니다. 이 예에서 0.1 과 0.10000038 간의 차이는 .000038%로, 사실상 0 이나 마찬가지입니다. 대부분의 연산에서 이처럼 극히 작은 부정확도는 중요하지 않습니다. 좀 더 넓게 생각하면, 부동 소수점 값을 아날로그 출력 모듈로 보내는 경우 .000038%의 차이가 있는 모듈로 보내는 값에 대한 출력 전압에는 차이가 없습니다.

부동 소수점 산술 연산에 대한 지침

다음 지침을 따르십시오.

특정 부동 소수점 산술 연산을 수행하는 경우 반올림 에러로 인해 정밀도가 손실될 수 있습니다. 부동 소수점 프로세서에는 결과 값에 영향을 미칠 수 있는 자체적인 내부 정밀도가 있습니다.

화폐 값 또는 총계 산출 기능에는 부동 소수점 산술을 사용하지 마십시오. INT 또는 DINT 값을 사용하고, 값을 위로 조정하고, 소수 자릿수를 추적합니다(또는 달러에는 INT 또는 DINT 값 중 하나를, 센트에는 INT 또는 DINT 중 나머지 하나를 사용).

부동 소수점 수를 비교하지 마십시오. 대신 범위 내에서 값을 확인합니다. 이를 위해 특히 LIM 명령어가 제공됩니다.

총계 산출의 예

REAL 데이터 유형의 정밀도는 총계 산출 응용 분야에 영향을 미치므로 매우 큰 숫자에 매우 작은 숫자를 더하는 경우 에러가 발생합니다.

예를 들어, 일정 기간 동안 어떤 숫자에 1 을 더합니다. 어느 시점이 되면 총 합계가 1 보다 훨씬 커지고 전체 결과를 저장할 충분한 비트가 없기 때문에 더하기가 더 이상 결과에 영향을 미치지 않습니다. 더하기는 상위 비트를 가능한 한 많이 저장하고 남은 하위 비트는 버립니다.

이 문제를 해결하기 위해 결과가 커질 때까지 작은 숫자에 대한 연산을 수행합니다. 그런 다음 추가 큰 수 연산을 위해 결과를 다른 위치로 보냅니다. 예:

- x 는 작은 증분 변수입니다.
- y 는 큰 증분 변수입니다.

- z 는 어디에서든 사용할 수 있는 현재 총 카운트입니다.
- x = x+1;
- if x = 100,000;
- {
- y = y + 100,000;
- x = 0;
- }
- z = y + x;

또 다른 예:

- x = x + some_tiny_number;
- if (x >= 100)
- {
- z = z + 100;
- x = x - 100; // there might be a tiny remainder
- }

배열을 통한 인덱스

로직에서 참조하는 배열 요소를 동적으로 변경하려면 태그 또는 식을 첨자로 사용하여 해당 요소를 가리킵니다. 이는 PLC-5 로직에서의 간접 주소 지정과 유사합니다. 식에서 다음 연산자를 사용하여 배열 첨자를 지정합니다.

팁:
- Logix Designer 에서는 데이터 유형 태그만 확장된 첨자를 허용하며 데이터 유형이 확장된 첨자 식은 지원하지 않습니다.
- 사용 가능한 모든 정수 요소 데이터 유형을 첨자 인덱스로 사용할 수 있습니다. 연산자에 SINT, INT, DINT 태그만 사용하여 첨자 식을 만듭니다.

연산자	설명
+	더하기
-	빼기/부정
*	곱하기
/	나누기

연산자	설명
논리곱	논리곱
FRD	BCD -> 정수
NOT	보수
또는	또는
TOD	정수 -> BCD
SQR	제곱근
XOR	배타적 논리합

예:

정의	예	설명
my_list DINT[10]으로 정의됨	my_list[5]	이 예는 배열의 요소 5 를 참조합니다. 첨자 값이 일정하게 유지되므로 이러한 참조는 정적입니다.
my_list DINT[10]으로 정의됨 위치 DINT 로 정의됨	값 5 를 위치로 MOV my_list[position]	이 예는 배열의 요소 5 를 참조합니다. 이러한 참조는 위치의 값을 변경하여 로직이 첨자를 변경할 수 있기 때문에 동적입니다.
my_list DINT[10]으로 정의됨 위치 DINT 로 정의됨 오프셋 DINT 로 정의됨	값 2 를 위치로 MOV 값 5 를 오프셋으로 MOV my_list[position+offset]	이 예는 배열의 요소 7(2+5)을 참조합니다. 이러한 참조는 위치 또는 오프셋의 값을 변경하여 로직이 첨자를 변경할 수 있기 때문에 동적입니다.

팁: 배열 첨자를 입력할 경우 입력한 배열 첨자가 지정한 배열의 경계 내에 있어야 합니다. 배열을 요소의 컬렉션으로 보는 명령어의 경우 첨자가 해당하는 차원을 벗어나면 메이저 폴트(유형 4, 코드 20)가 발생합니다.

비트 주소 지정

비트 주소 지정은 큰 컨테이너 내의 특정 비트에 액세스하는 데 사용됩니다. 큰 컨테이너에는 정수, 구조 또는 BOOL 배열이 포함됩니다. 예:

정의	예	설명
Variable0 LINT 로 정의됨 64 비트가 있음	variable0.42	이 예는 variable0 의 비트 42 를 참조합니다.
variable1 DINT 로 정의됨 32 비트 보유	variable1.2	이 예는 variable1 의 비트 2 를 참조합니다.
variable2 INT 로 정의됨 16 비트 보유	variable2.15	이 예는 variable2 의 비트 15 를 참조합니다.
variable3 SINT 로 정의됨 8 비트 보유	variable3.[4]	이 예는 variable3 의 비트 4 를 참조합니다.
variable4 COUNTER 구조로 정의됨 5 개의 상태 비트 보유	variable4.DN	이 예는 variable4 의 DN 비트를 참조합니다.
MyVariable BOOL[100]로 정의됨 MyIndex SINT 로 정의됨	MyVariable[(MyIndex AND NOT 7) / 8].[MyIndex AND 7]	이 예는 BOOL 배열 내의 비트를 참조합니다.
BOOL[20]로 정의된 MyArray	MyArray[3]	이 예는 MyArray 의 비트 3 을 참조합니다.
variable5 ULINT 로 정의됨 64 비트 보유	variable5.53	이 예는 variable5 의 비트 53 을 참조합니다.

BOOL 유형 태그가 허용되는 곳이라면 어디서나 비트 주소 지정을 사용합니다.

추가 참조

배열을 통한 인덱스 페이지의 978

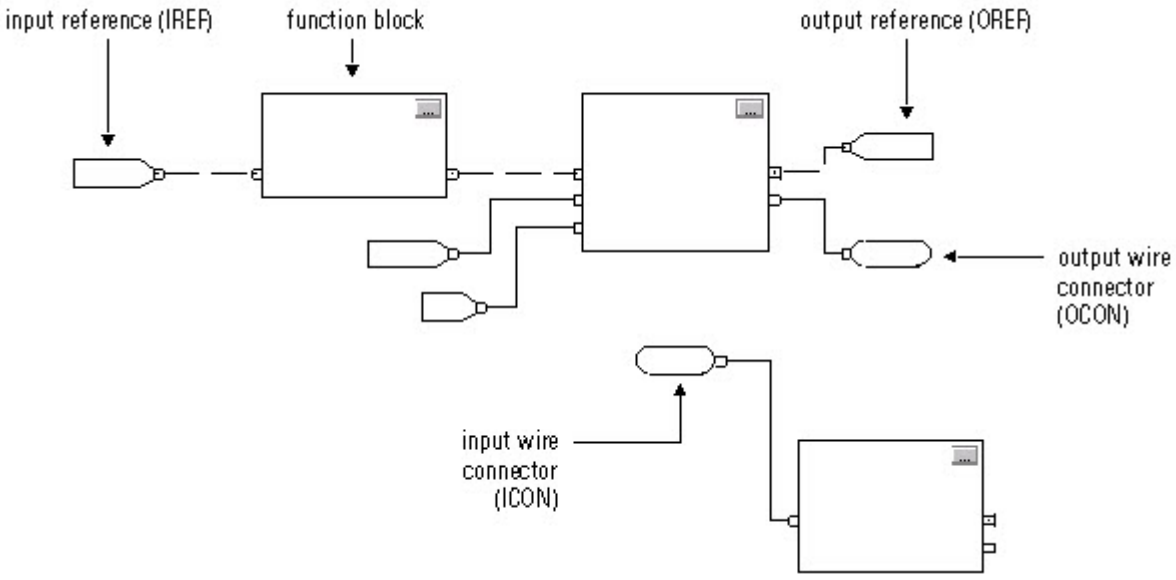
펑션 블록 속성

펑션 블록 프로그래밍에 고유한 문제에 대한 자세한 내용을 보려면 아래 항목을 클릭하십시오. 펑션 블록 루틴의 작동 방식을 파악하려면 이 정보를 살펴보십시오.

추가 참조

펑션 블록 요소 선택

장치를 제어하려면 다음 요소를 사용합니다.



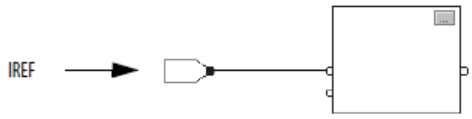
다음 표를 이용해 펑션 블록 요소를 선택할 수 있습니다.

입력 장치나 태그에서 값을 제공하려는 경우	입력 참조(IREF) 사용
값을 출력 장치나 태그로 전송	출력 참조(OREF)
입력 값(들)에 대한 연산을 수행하고 출력 값(들) 생성.	펑션 블록
다음의 경우 펑션 블록 간 데이터 전송 • 동일 시트에서 떨어져 있는 경우 • 동일 루틴 내에서 서로 다른 시트에 있는 경우	출력 와이어 커넥터(OCON)와 입력 와이어 커넥터(ICON)
루틴 내 여러 지점으로 데이터 분산	단일 출력 와이어 커넥터(OCON)와 여러 입력 와이어 커넥터(ICON)

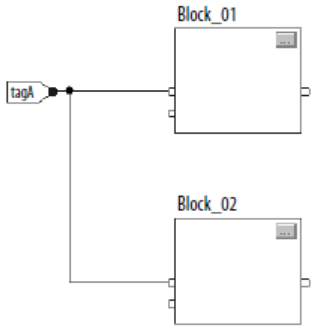
펑션 블록은 입력 참조를 블록 구조로 옮깁니다. 필요하면 펑션 블록이 입력 참조를 REAL 값으로 변환합니다. 펑션 블록은 실행하고 결과를 출력 참조로 보냅니다. 또 필요하면 펑션 블록이 이 결과 값을 REAL 에서 출력 참조용 데이터 형식으로 변환합니다.

데이터 래칭

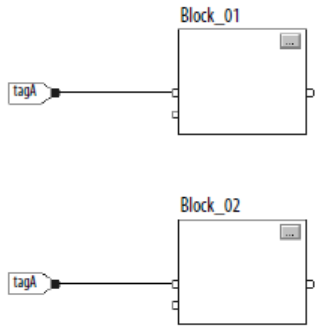
IREF 를 이용해 펑션 블록 명령어에 입력 데이터를 지정하면 IREF 의 데이터는 그 펑션 블록 루틴의 스캔에 맞춰 래칭됩니다. IREF 는 프로그램 범위와 컨트롤러 범위 태그의 데이터를 래칭합니다. 컨트롤러는 매 스캔 시작 시점에 IREF 데이터를 모두 업데이트합니다.



이 보기에서 tagA 의 값은 루틴 실행 시작 시점에 저장됩니다. 저장된 값은 Block_01 이 실행될 때 사용됩니다. 같은 값은 Block_02 가 실행될 때에도 사용됩니다. 루틴 실행 중 tagA 의 값이 변하면 IREF 에 저장된 tagA 의 값은 다음 루틴 실행 때까지 변하지 않습니다.

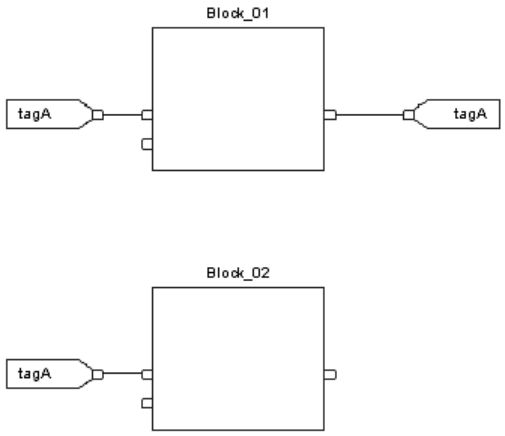


이 예는 위와 동일합니다. tagA 의 값은 루틴 실행 시작 시점에 한 번만 저장됩니다. 루틴은 이 저장된 값을 루틴 내내 이용합니다.



같은 루틴의 여러 IREF 와 OREF 에서 같은 태그를 이용할 수 있습니다. IREF 의 태그 값은 루틴에서 스캔마다 래칭되므로 OREF 가 루틴 실행 중 다른 값을 획득하더라도 IREF 는 모두 같은 값을 이용합니다.

이 보기에서 루틴이 이 스캔을 시작할 때 tagA 의 값이 25.4 이고 Block_01 이 tagA 의 값을 50.9 로 바꾸면 Block_02 로 연결된 두 번째 IREF 는 Block_02 가 이 스캔을 실행할 때 25.4 를 계속 사용합니다. tagA 값 50.9 는 다음 스캔이 시작될 때까지 이 루틴에서 어떤 IREF 도 사용하지 않습니다.



실행 순서

Logix Designer 프로그래밍 응용 프로그램은 다음의 경우 루틴의 펑션 블록에 대한 실행 순서를 자동으로 결정합니다.

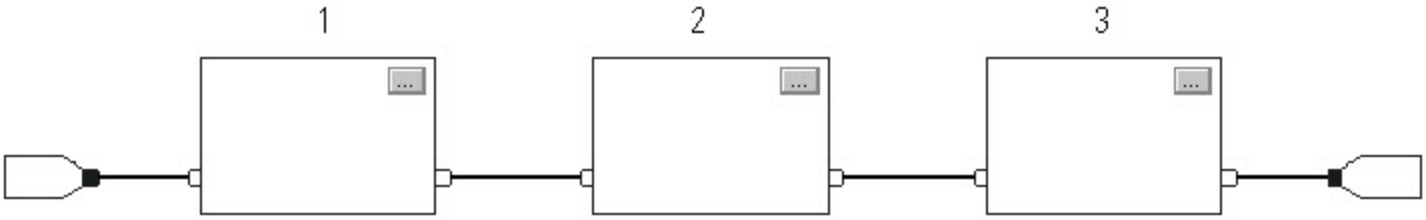
- 펑션 블록 루틴 확인
- 펑션 블록 루틴을 포함하는 프로젝트 확인
- 펑션 블록 루틴을 포함하는 프로젝트 다운로드

펑션 블록을 함께 연결하고 필요할 경우 피드백 와이어의 데이터 흐름을 지정하여 실행 순서를 정의할 수 있습니다.

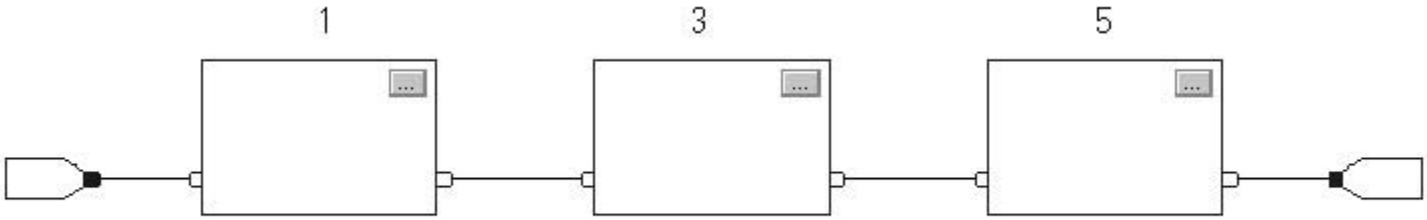
펑션 블록이 함께 연결되지 않는 경우 어느 블록이 첫 번째로 실행는 것이 중요하지 않습니다. 블록 간에 데이터 흐름이 없는 경우

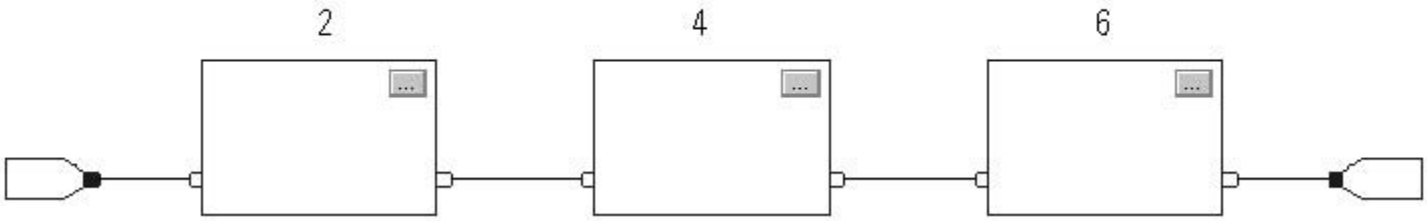
블록을 순차적으로 연결하면 실행 순서가 입력에서 출력으로 이동합니다. 컨트롤러에서 해당 블록을 실행하기 전에 먼저 블록의 입력에 사용 가능한 데이터가 있어야 합니다. 예를 들면,

블록 2 의 출력이 블록 3 의 입력을 제공하므로 블록 2 를 블록 3 보다 먼저 실행해야 합니다.



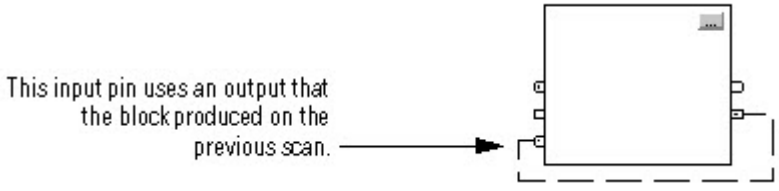
실행 순서는 함께 연결되는 블록하고만 관련이 있습니다. 다음 예에서 두 블록 그룹이 함께 연결되지 않았으므로 해당되지 않습니다. 특정 그룹 내 블록은 해당 그룹 내의 다른 블록과 관련하여 적절한 순서대로 실행됩니다.



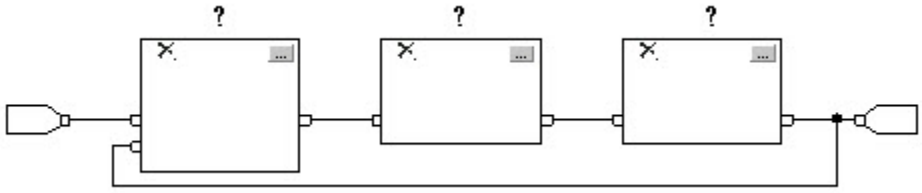


루프 분석

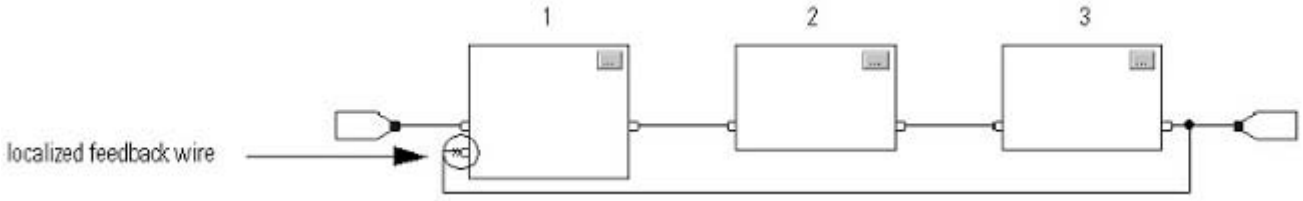
블록 주변에서 피드백 루프를 생성하려면 블록의 출력 핀을 동일 블록의 입력 핀에 연결합니다. 다음 예는 정상입니다. 루프에 단일 블록만 포함되므로 실행 순서가 중요하지 않습니다.



블록 그룹이 루프 안에 있으면 컨트롤러는 처음 실행할 블록을 결정할 수 없습니다. 다시 말해서 루프를 분석할 수 없습니다.

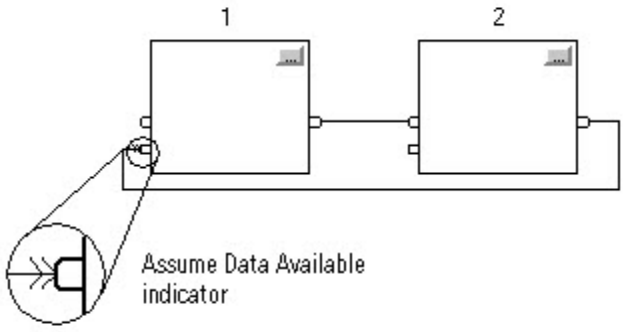
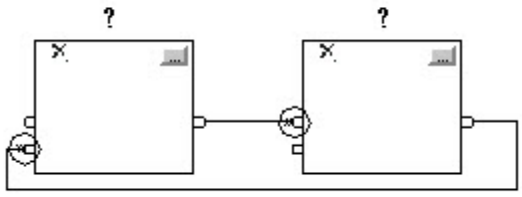


처음 실행할 블록을 식별하려면 *사용 가능한 데이터로 가정*(Assume Data Available) 표시기를 사용하여 루프를 생성하는 입력 와이어(피드백 와이어)를 표시합니다. 다음 예에서는 블록 1 에 루틴의 이전 실행에서 생성된 블록 3 의 출력이 사용됩니다.



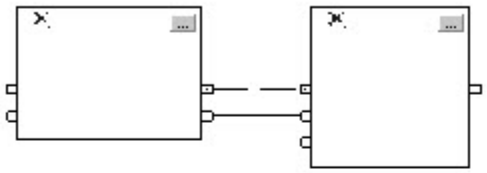
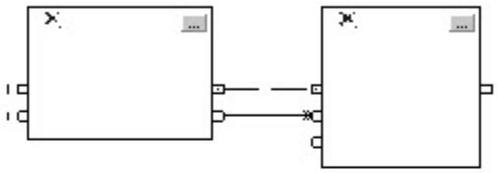
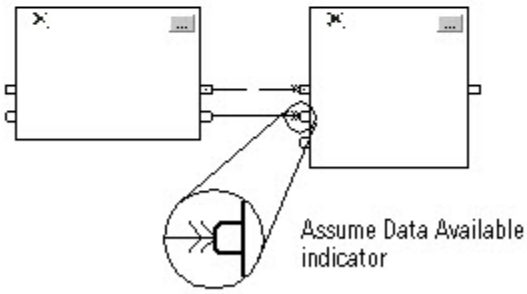
사용 가능한 데이터로 가정(Assume Data Available) 표시기는 루프 내에서 데이터 흐름을 정의합니다. 화살표는 데이터가 루프의 첫 번째 블록에 입력으로 제공됨을 나타냅니다.

사용 가능한 데이터로 가정(Assume Data Available) 표시기를 사용하여 루프의 모든 와이어를 표시하지 *마십시오*.

This is OK	This is NOT OK
 The *Assume Data Available* indicator defines the data flow within the loop.	 The controller cannot resolve the loop because all the wires use the *Assume Data Available* indicator.

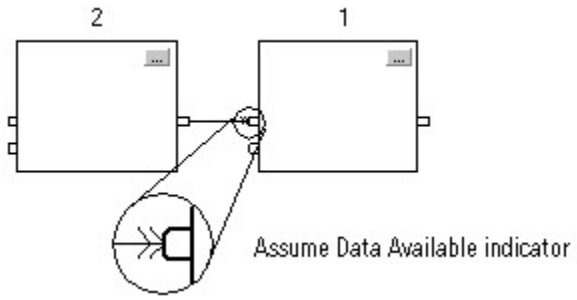
두 블록 사이 데이터 흐름 분석

두 개 이상의 와이어를 사용하여 두 블록을 연결하는 경우 두 블록 사이에 있는 모든 와이어에 대해 동일한 데이터 흐름 표시기를 사용합니다.

This is OK	This is NOT OK
 Neither wire uses the *Assume Data Available* indicator.	 One wire uses the *Assume Data Available* indicator while the other wire does not.
 Both wires use the *Assume Data Available* indicator.	

1 회 스캔 지연 생성

블록 사이에 1 회 스캔 지연을 생성하려면 사용 가능한 데이터로 가정(Assume Data Available) 표시기를 사용합니다. 다음 예에서는 블록 1 이 첫 번째로 실행됩니다. 루틴의 이전 스캔에서 생성된 블록 2 의 출력이 사용됩니다.



요약

요약하자면, 펑션 블록 루틴은 다음 순서대로 실행됩니다.

1. 컨트롤러가 IREF 의 모든 데이터 값을 래치합니다.
2. 컨트롤러가 연결 방식에 따라 결정된 순서대로 다른 펑션 블록을 실행합니다.
3. 컨트롤러가 출력을 OREF 에 씁니다.

오버플로우 상태에 대한 펑션 블록의 응답

일반적으로 이력을 간직한 펑션 블록 명령어는 오버플로우 발생 시 ±NAN 또는 ±INF 값으로 이력을 업데이트하지 않습니다. 각 명령어는 오버플로우 상태에 대해 아래 응답 중 하나를 지닙니다.

응답	명령어
응답 1 블록은 알고리즘을 실행하고 ±NAN 또는 ±INF. ±NAN 또는 ±INF 이면 블록은 ±NAN 또는 ±INF.	ALM NTCH DEDT PMUL DERV POSP ESEL RLIM FGEN RMPS HPF SCRV LDL2 SEL LDLG SNEG LPF SRTP MAVE SSUM MAXC TOT MINC UPDN MSTD MUX
응답 2 출력 제한이 있는 블록은 알고리즘을 실행하고 ±NAN 또는 ±INF 의 결과를 확인합니다. 출력 한도는 HighLimit 와 LowLimit 입력 파라미터로 정의됩니다. ±INF 이면 블록은 제한된 결과를 출력합니다. ±NAN 이면 출력 한도는 사용되지 않고 블록은 ±NAN 을 출력합니다.	HLL, INTG, PI, PIDE, SCL, SOC
응답 3 오버플로우 조건은 적용되지 않습니다. 아래 명령어에는 일반적으로 부울 출력이 있습니다.	BAND, BNOT, BOR, BXOR, CUTD, D2SD, D3SD, DFF, JKFF, OSFI, OSRI, RESD, RTOR, SETD, TOFR, TONR

타이밍 모드

이 프로세스 제어 및 드라이브 명령어는 여러 가지 타이밍 모드를 지원합니다.

- DEDT
- DERV
- HPF
- INTG
- LDL2
- LDLG
- LPF
- NTCH
- PI
- PIDE
- RLIM
- SCRV
- SOC
- TOT

세 가지 타이밍 모드가 있습니다.

타이밍 모드	설명	
주기	주기 모드는 기본 모드로 대부분이 제어 용도에 적합합니다. 이 모드를 이용하는 명령어를 주기 태스크를 실행하는 루틴에 놓기를 권장합니다. 이 명령어의 델타 시간(DeltaT)은 다음과 같이 결정됩니다.	
	명령어 실행 위치	**DeltaT=**
	주기 태스크	태스크의 기간
	이벤트 또는 연속 태스크	이전 실행 이후 경과 시간 컨트롤러는 경과 시간을 소수점 이하 밀리초 단위(ms)를 절사합니다. 여컨대, 경과 시간이 10.5 ms 라면 컨트롤러는 DeltaT=10 ms 로 설정합니다.
	프로세스 입력의 업데이트는 태스크의 실행과 동기화하거나 태스크 실행보다 5 ~ 10 배 빠르게 샘플링해야 합니다. 입력과 명령어 간 샘플링 에러를 최소화하기 위함입니다.	
오버 샘플링	오버 샘플링 모드에서는 명령어가 사용하는 델타 시간(DeltaT)이 명령어의 OversampleDT 파라미터에 적힌 값이 됩니다. 프로세스 입력에 타임 스탬프 값이 있다면 대신 실시간 샘플링 모드를 사용합니다. 명령어가 실행될 때 제어할 프로그램에 로직을 추가하십시오. 예를 들어, OversampleDeltaT 값으로 설정된 타이머와 명령어의 EnableIn 입력을 이용해 실행을 제어할 수 있습니다. 프로세스 입력은 명령어의 실행보다 5 ~ 10 배 빠르게 샘플링해야 합니다. 입력과 명령어 간 샘플링 에러를 최소화하기 위함입니다.	

실시간 샘플링	실시간 샘플링 모드에서는 명령어가 사용하는 델타 시간(DeltaT)이 프로세스 입력이 업데이트에 상응하는 두 타임 스탬프 값의 차가 됩니다. 프로세스 입력에 업데이트와 관련된 타임 스탬프가 있고 정밀한 조정이 필요할 때 이 모드를 이용합니다. 타임 스탬프 값은 명령어의 RTSTimeStamp 파라미터에 입력된 태그 이름에서 읽힙니다. 일반적으로 이 태그 이름은 프로세스 입력과 관련된 입력 모듈에 표시된 파라미터가 됩니다. 이 명령어는 구성된 RTSTime 값(예상 업데이트 기간)과 계산된 DeltaT 를 비교해 프로세스 입력의 모든 업데이트가 명령어에 읽힐 준비가 되었는지 판단합니다. DeltaT 가 구성 시간의 1 밀리초 이내가 아니라면 명령어는 RTSMissed 상태 비트를 설정해 모듈에서 입력 업데이트를 읽는 데 문제가 있다고 표시합니다.

시간 기반 명령어에 DeltaT 의 상수값이 있어야 제어 알고리즘이 프로세스 출력을 제대로 계산합니다. DeltaT 가 가변적이면 프로세스 출력에 비연속이 발생합니다. 비연속의 심각도는 명령어, 그리고 DeltaT 가 변하는 정도에 따라 달라집니다.

비연속이 발생하는 결과는 다음과 같습니다.

- 명령어가 스캔 중에 실행되지 않습니다.
- 명령어가 태스크 중에 여러 번 실행됩니다.
- 태스크가 실행 중이고 태스크 실행 속도 또는 프로세스 입력의 샘플 시간이 변합니다.
- 사용자는 태스크 실행 중 시간 기반 모드를 변경합니다.
- 태스크 실행 중 필터 블록에서 Order 파라미터가 바뀝니다.
- Order 파라미터를 바꾸면 명령어에서 다른 제어 알고리즘이 선택됩니다.

타이밍 모드의 공통 명령어 파라미터

시간 기반 모드를 지원하는 명령어는 아래와 같은 입력 및 출력 파라미터가 있습니다.

입력 파라미터

입력 파라미터	데이터 유형	설명
TimingMode	DINT	타이밍 실행 모드 선택. 값: 설명: 0 주기 모드 1 오버 샘플링 모드 2 실시간 샘플링 모드 유효값 = 0 ~ 2 기본값 = 0 TimingMode = 0 이고 태스크가 주기적이면 주기 타이밍이 활성화되고 DeltaT 는 태스크 스캔 속도로 설정됩니다. TimingMode = 0 이고 태스크가 이벤트 또는 연속이면 주기 타이밍이 활성화되고 DeltaT 는 명령어가 마지막으로 실행된 후 경과 시간 범위와 동일하게 설정됩니다. TimingMode = 1 이면 오버 샘플링 타이밍이 활성화되고 DeltaT 는 OversampleDT 파라미터의 값으로 설정됩니다. TimingMode = 2 이면 실시간 샘플링 타이밍이 활성화되고 DeltaT 는 입력과 관련된 모듈에서 읽은 현재 및 이전 타임 스탬프 값 간의 차가 됩니다. TimingMode 가 유효하지 않으면, 명령어는 Status 에서 적당한 비트를 설정합니다.
OversampleDT	REAL	오버 샘플링 타이밍의 실행 시간 DeltaT 에 사용되는 값은 초 단위입니다. TimingMode = 1 이면, OversampleDT = 0.0 는 제어 알고리즘의 실행을 중지합니다. 유효하지 않으면 명령어는 DeltaT = 0.0 를 설정하고 Status 에 적절한 비트를 설정합니다. 값 = 0~4194.303 초 기본값 = 0.0
RTSTime	DINT	실시간 샘플링 타이밍의 모듈 업데이트 기간. 예상 DeltaT 업데이트 기간은 단위가 밀리초입니다. 업데이트 기간은 일반적으로 모듈의 업데이트 시간을 구성할 때 사용되는 값이 됩니다. 유효하지 않으면, 명령어는 Status 에서 적절한 비트를 설정하고 RTSMissed 검사를 중단합니다. 유효값 = 1 ~ 32,767 ms 기본값 = 1

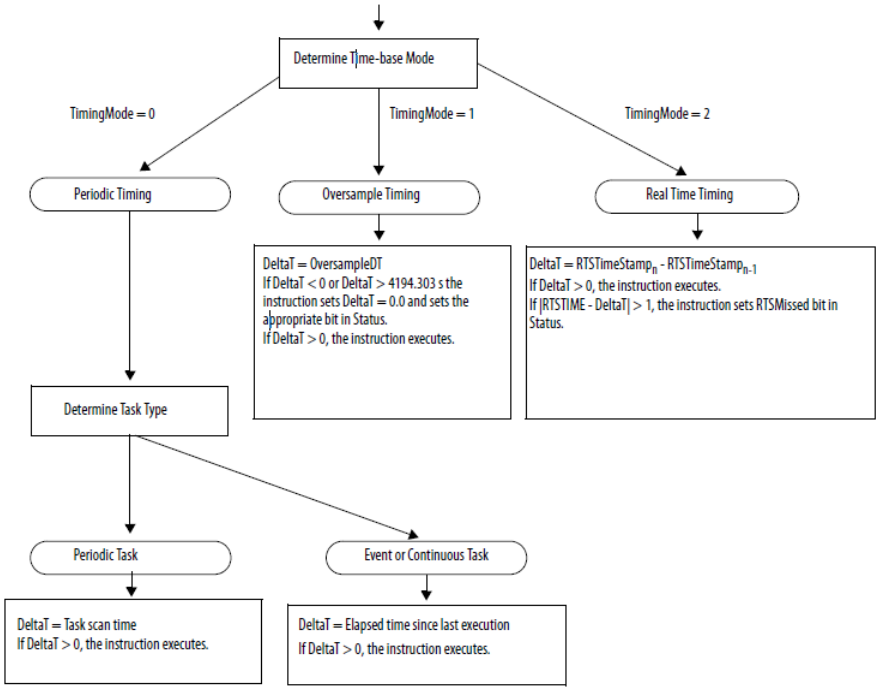
RTSTimeStamp	DINT	실시간 샘플링 타이밍의 모듈 시간 스탬프 값. 입력 신호의 마지막 업데이트에 상응하는 타임 스탬프 값. 이 값은 DeltaT 계산에 사용됩니다. 유효하지 않으면, 명령어는 Status 에서 적절한 비트를 설정하고 제어 알고리즘의 실행을 중단하고 RTSMissed 검사를 중단합니다. 유효값 =0 ~ 32,767 ms (32767 에서 0 으로 건너뛰기) 1 카운트 = 1 밀리초 기본값 = 0

출력 파라미터

출력 파라미터	데이터 유형	설명
DeltaT	REAL	업데이트 간 경과 시간. 프로세스 출력을 계산하는 데 제어 알고리즘에 필요한 경과 시간(초)입니다. 주기: DeltaT = 태스크가 정기이면 태스크 스캔 속도, 태스크가 이벤트 또는 연속이면 DeltaT = 이전 명령 실행 후 경과 시간 오버 샘플링: DeltaT = OversampleDT 실시간 샘플링: DeltaT = (RTSTimeStampn - RTSTimeStampn-1)
Status	DINT	펑션 블록의 상태.
TimingModeInv (Status.27)	BOOL	유효하지 않은 TimingMode 값.
RTSMissed (Status.28)	BOOL	실시간 샘플링 모드에서만 사용. ABS | DeltaT - RTSTime | > 1 (.001 초)일 때 설정.
RTSTimeInv (Status.29)	BOOL	유효하지 않은 RTSTime 값.
RTSTimeStampInv (Status.30)	BOOL	유효하지 않은 RTSTimeStamp 값.
DeltaTInv (Status.31)	BOOL	유효하지 않은 DeltaT 값.

타이밍 모듈의 개요

다음 다이어그램은 명령어가 적절한 타이밍 모드를 결정하는 원리를 보여줍니다.



프로그램/작업자 제어

아래 명령어들은 프로그램/작업자 제어의 개념을 지원합니다.

- 고급 선택(ESEL)
- 총계 계산기(TOT)
- 향상된 PID(PIDE)
- 램프/소크(RMPS)
- 이산 2-상태 장치(D2SD)
- 이산 3-상태 장치(D3SD)

프로그램/작업자 제어를 통해 사용자 프로그램 및 작업자 인터페이스 장치 양쪽에서 동시에 이 명령어들을 제어할 수 있습니다. 명령어가 프로그램 제어에 있을 때는 명령어에 대한 프로그램 입력으로, 작업자 제어에 있을 때는 명령어에 대한 작업자 입력으로 제어합니다.

프로그램 또는 작업자 제어는 아래 입력을 사용하여 결정됩니다.

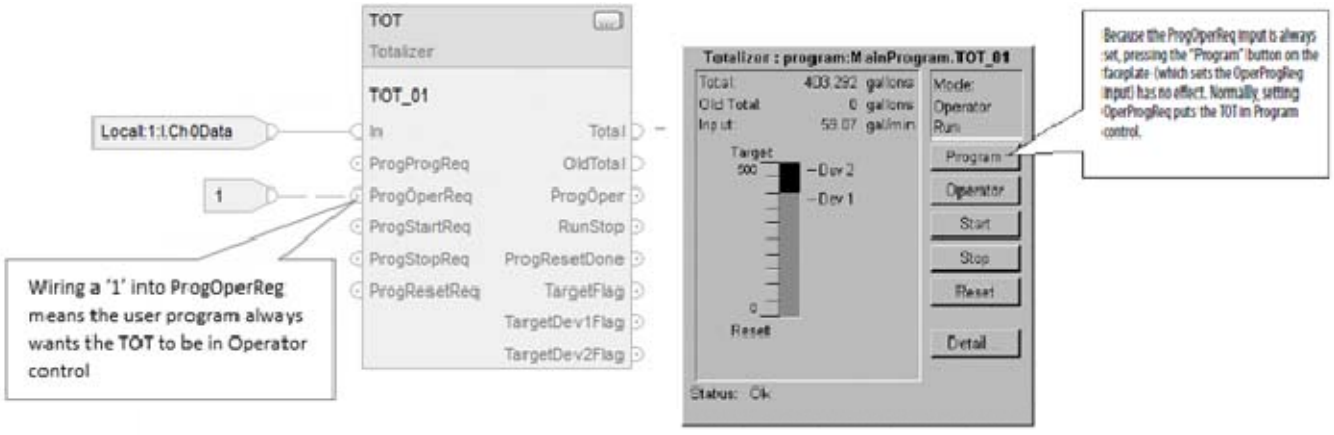
입력	설명
.ProgProgReq	프로그램 제어로 전환되는 프로그램 요청입니다.
.ProgOperReq	작업자 제어로 전환되는 프로그램 요청입니다.
.OperProgReq	프로그램 제어로 전환되는 작업자 요청입니다.
.OperOperReq	작업자 제어로 전환되는 작업자 요청입니다.

명령어가 프로그램 제어에 있는지 아니면 작업자 제어에 있는지를 결정하려면 ProgOper 출력을 확인하면 됩니다. ProgOper 가 설정되어 있으면 명령어는 프로그램 제어에 있고, ProgOper 가 해제되면 명령어는 작업자 제어에 있습니다.

두 입력 요청 비트가 모두 설정되어 작업자 제어가 프로그램 제어보다 우선합니다. 예를 들어, ProgProgReq 와 ProgOperReq 가 모두 설정되어 있으면 명령어는 작업자 제어로 실행됩니다.

프로그램 요청 입력은 작업자 요청 입력보다 우선합니다. 이에 따라 ProgProgReq 및 ProgOperReq 입력을 사용하여 명령어를 원하는 제어에 '잠글' 수 있는 능력이 제공됩니다.

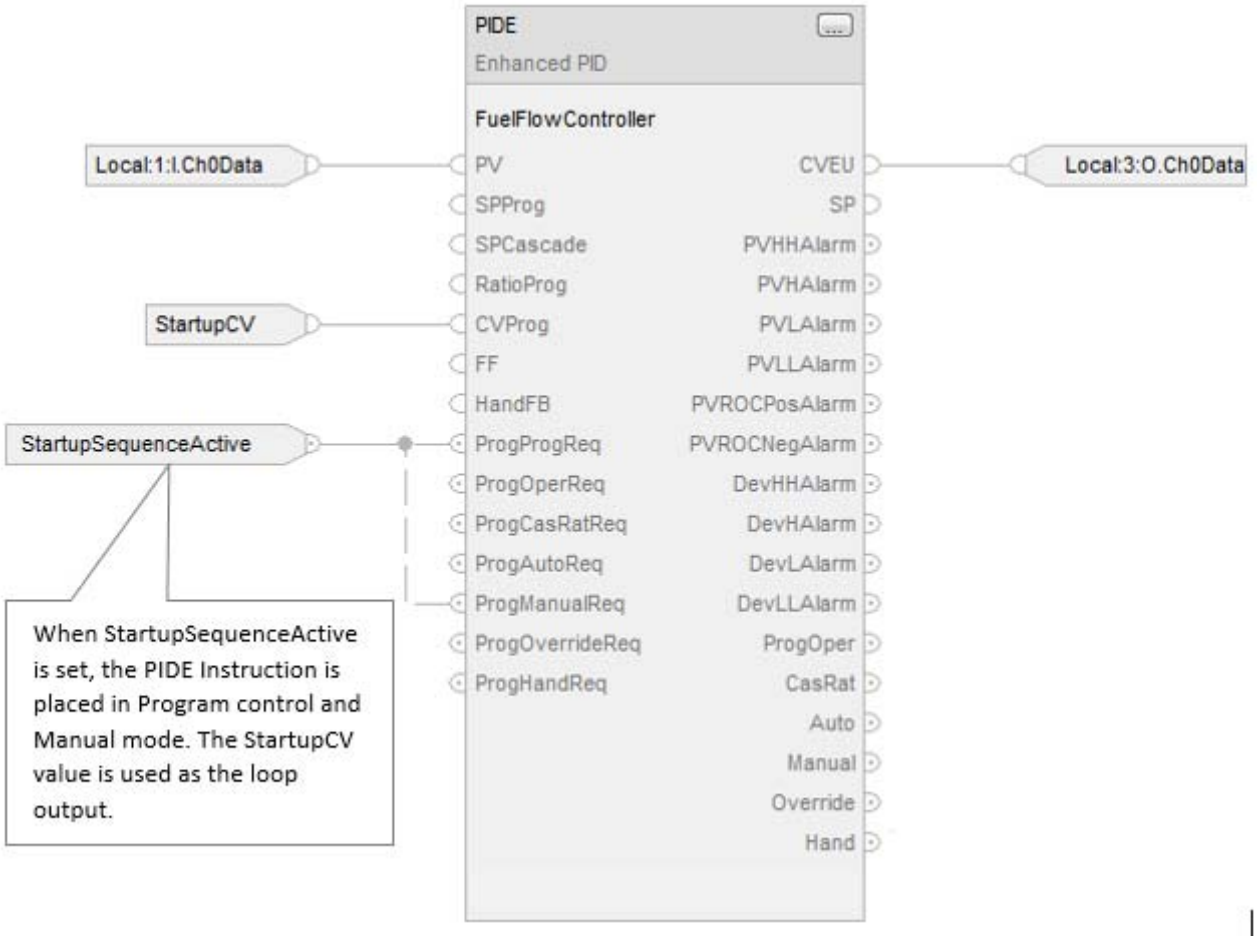
예를 들어, 총계 계산 명령어는 항상 작업자 제어에 있고 사용자 프로그램은 이 장치의 실행 또는 정지를 결코 제어하지 않는다고 가정합시다. 이 경우, 리터럴 값 1 을 ProgOperReq 로 전송할 수 있습니다. 이렇게 하면 작업자는 작업자 인터페이스 장치에서 OperProgReq 를 설정해서 총계 계산기를 프로그램 제어로 이동시키는 작업을 절대로 할 수 없게 됩니다.



마찬가지로 ProgProgReq 를 상시 설정하면 명령어를 프로그램 제어 안에 '잠글' 수 있습니다. 이는 작업자가 실수로 명령어를 제어하는 것을 걱정할 필요가 없이 프로그램이 명령어의 조치를 제어하게 만들고자 하는 경우에 자동 시작 시퀀스에 유용합니다.

이 예에서는 프로그램이 시작 작업 중에 ProgProgReq 입력을 설정한 다음에 시작 작업이 완료되면 ProgProgReq 입력을 해제합니다. ProgProgReq 입력이 해제되면 명령어는 변경 요청을 수신할 때까지 계속 프로그램 제어에 유지됩니다. 예를 들어, 작업자가 면판에서 해당 명령어의 제어권 취득을 위해 OperOperReq 입력을 설정할 수 있습니다.

아래 예는 명령어를 프로그램 제어 내에 잠그는 과정을 보여주고 있습니다.

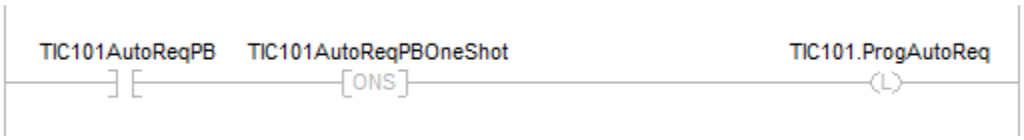


명령에 대한 작업자 요청 입력이 있을 때마다 항상 이 명령이 실행되면서 입력을 지웁니다. 이를 통해 작업자 인터페이스는 단지 원하는 모드 요청 비트를 설정하는 것만으로도 이들 명령어의 작업을 수행할 수 있습니다. 요청 비트를 재설정하도록 작업자 인터페이스를 프로그래밍할 필요가 없습니다. 예를 들어, 작업자 인터페이스가 PIDE 명령어에 대한 OperAutoReq 입력을 설정하는 경우, PIDE 명령어가 실행되면서 적절한 응답이 무엇인지 결정하고 OperAutoReq 를 지웁니다.

프로그램 요청 입력은 일반적으로 해당 명령어에 대한 입력으로 전송되기 때문에 보통은 해당 명령어에 의해 해제되지 않습니다. 명령어가 이들 입력을 지우면 전송된 입력이 입력을 다시 설정합니다. 해당 명령어가 프로그램 요청을 삭제하는 방식으로 다른 로직을 사용하여 프로그램 요청을 설정하려는 경우도 있을 수 있습니다. 이 경우에는 ProgValueReset 입력을 설정할 수 있으며, 명령어가 실행될 때마다 프로그램 모드 요청 입력을 지웁니다.

이 예에서는 다른 루틴에 있는 래더 로직의 렁을 사용하여 푸시 단추를 누를 때 ProgAutoReq 를 PIDE 명령어에 원샷 래칭합니다.

TIC101AutoReq 푸시 단추를 누르면 PIDE 명령어 TIC101 에 대해 ProgAutoReq 를 원샷 래칭합니다. TIC101 은 ProgValueReset 입력 세트를 사용하여 구성되었습니다. ProgAutoReq 는 ProgValuieReset 이 항상 설정되기 때문에 리셋됩니다.



ST(스트럭쳐드 텍스트) 프로그래밍

다음은 ST(스트럭쳐드 텍스트) 프로그래밍에 고유한 문제입니다. ST(스트럭쳐드 텍스트) 프로그래밍의 실행 과정을 확실히 이해하도록 다음 주제를 검토하십시오.

ST(스트럭쳐드 텍스트) 구문

ST(스트럭쳐드 텍스트)는 실행할 항목을 정의하기 위해 문을 사용하는 텍스트 프로그래밍 언어입니다.

- ST(스트럭쳐드 텍스트)는 대/소문자를 구분하지 않습니다.
- 탭과 캐리지 리턴(다른 행)을 사용해 ST(스트럭쳐드 텍스트)를 읽기 쉽게 표현합니다. ST(스트럭쳐드 텍스트)의 실행에는 영향을 주지 않습니다.

ST(스트럭쳐드 텍스트)는 대/소문자를 구분하지 않습니다.
ST(스트럭쳐드 텍스트)는 다음 구성 요소를 포함할 수 있습니다.

용어	정의	예제
할당	할당 명령문을 사용하여 값을 태그에 할당합니다. := 연산자는 할당 연산자입니다. 할당을 세미콜론 ';'으로 종료합니다.	태그:= 식;
Expression	식은 전체 할당 또는 구성 명령문의 일부입니다. 식은 숫자(수식), 문자열(문자열 식), 또는 참 또는 거짓 상태(BOOL 식)으로 계산됩니다.	
태그 식	데이터가 저장되는 메모리의 명명된 영역입니다(BOOL, SINT, INT, DINT, REAL, String).	value1
직접 식	상수 값	4
연산자 식	식 내에 연산을 지정하는 기호 또는 니모닉입니다.	tag1 + tag2 tag1 >= value1
함수 식	실행되면 함수가 값 1 개를 생성합니다. 함수의 피연산자를 포함하려면 괄호를 사용합니다. 구문이 유사하더라도 함수는 식에만 사용될 수 있는 점에서 명령어와 함수가 서로 다릅니다. 명령어는 식에 사용될 수 없습니다.	function(tag1)

명령어	명령어는 독립형 명령문입니다.
	명령어는 괄호를 사용하여 피연산자를 포함합니다.
	명령어에 따라 피연산자가 0 개, 1 개 또는 여러 개 있을 수 있습니다.
	실행되면 명령어는 데이터 구조의 일부인 하나 이상의 값을 생성합니다. 명령어를 세미콜론 ';'으로 종료합니다.
	구문이 유사하더라도 명령어는 식에 사용될 수 없는 점에서 명령어와 함수가 서로 다릅니다. 함수는 식에만 사용될 수 있습니다.
		instruction();
		
		instruction(operand);
		
		instruction(operand1, operand2,operand3);

구성	구조화된 ST(스트럭쳐드 텍스트)(즉, 다른 명령문)를 트리거하는 데 사용하는 조건 명령문입니다. 구성을 세미콜론 ';'으로 종료합니다.	IF...THEN CASE FOR...DO WHILE...DO REPEAT...UNTIL EXIT

주석	ST(스트럭쳐드 텍스트)의 섹션이 수행하는 기능을 설명하거나 구체화하는 텍스트입니다. ST(스트럭쳐드 텍스트)를 해석하기 쉽도록 주석을 사용합니다. 주석은 ST(스트럭쳐드 텍스트)의 실행에는 영향을 주지 않습니다. 주석은 ST(스트럭쳐드 텍스트)의 어디에나 나타날 수 있습니다.	//comment (*start of comment . . . end of comment*) /*start of comment . . . end of comment*/

추가 참조

ST(스트럭쳐드 텍스트) 구성 요소: 할당 페이지의 1000

ST(스트럭쳐드 텍스트) 구성 요소: 식 페이지의 1003

ST(스트럭쳐드 텍스트) 구성 요소: 명령어 페이지의 1010

ST(스트럭쳐드 텍스트) 구성 요소: 구성 페이지의 1011

ST(스트럭쳐드 텍스트) 구성 요소: 주석 페이지의 999

ST(스트럭쳐드 텍스트) 구성 요소: 주석

ST(스트럭쳐드 텍스트)를 해석하기 쉽도록 하려면 주석을 추가합니다.

- 주석을 사용하면 ST(스트럭쳐드 텍스트)가 작동하는 방법을 일반 언어로 설명할 수 있습니다.
- 주석은 ST(스트럭쳐드 텍스트)의 실행에는 영향을 주지 않습니다.

ST(스트럭쳐드 텍스트)에 주석을 추가하려면:

주석을 추가하려면	다음 형식 중 하나를 사용
단일 줄에	//comment (*comment*) /*comment*/
ST(스트럭쳐드 텍스트) 줄의 끝에	
ST(스트럭쳐드 텍스트) 줄 내에	(*comment*) /*comment*/
두 줄 이상에	(*start of comment. . .end of comment*) /*start of comment. . .end of comment*/

예:

Format	예

//comment	**줄의 시작에**
	//Check conveyor belt direction
	IF conveyor_direction THEN...
	줄의 끝에
	ELSE //If conveyor isn't moving, set alarm light
	light := 1;
	END_IF;

(*comment*)	Sugar.Inlet[:=]1;(*open the inlet*) IF Sugar.Low (*low level LS*)& Sugar.High (*high level LS*)THEN... (*Controls the speed of the recirculation pump. The speed depends on the temperature in the tank.*) IF tank.temp > 200 THEN...
/*comment*/	Sugar.Inlet:=0;/*close the inlet*/ IF bar_code=65 /*A*/ THEN... /*Gets the number of elements in the Inventory array and stores the value in the Inventory_Items tag*/ SIZE(Inventory,0,Inventory_Items);

ST(스트럭쳐드 텍스트) 구성 요소: 할당

할당을 사용하여 태그 내에 저장된 값을 변경합니다. 할당의 구문은 다음과 같습니다.

태그:= 식;

여기에서:

구성 요소	설명	
태그	새 값을 가져오는 태그를 나타내며, 태그는 BOOL, SINT, INT, DINT, STRING 또는 REAL 이어야 합니다. **팁:** STRING 태그는 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에만 적용할 수 있습니다.	
:=	할당 기호	
식	태그에 할당할 새 값을 나타냄	
	태그의 데이터 유형	**사용하는 식의 형식**
	BOOL	BOOL
	SINT INT DINT REAL	숫자

	STRING (CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러만 해당).	문자열 형식, 문자열 태그 및 문자열 리터럴 포함 (CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러만 해당).
;	할당을 종료함	

이 태그는 다른 할당이 값을 변경할 때까지 할당된 값을 보존합니다.

식은 즉시 값 또는 다른 태그 이름과 같이 단순할 수도 있고 복잡하여 여러 연산자 및 함수 또는 둘 다 포함할 수도 있습니다. 자세한 내용은 식을 참조하십시오.

팁: I/O 모듈 데이터는 비동기 방식으로 로직 실행으로 업데이트합니다. 입력을 로직에 여러 번 참조하면 입력은 별도의 참조 간에 상태를 변경할 수 있습니다. 입력이 각 참조에 대해 같은 상태를 가져야 하는 경우, 입력 값을 버퍼링하고 해당 버퍼 태그를 참조합니다. 자세한 내용은 LOGIX 5000 Controllers Common Procedures (발행 번호 1756-PM001)를 참조하십시오.

로직 실행 중에 데이터를 자동으로 버퍼링하는 입력 및 출력 프로그램 파라미터를 사용할 수도 있습니다. LOGIX 5000 Controllers Program Parameters Programming Manual (발행 번호 1756-PM021)을 참조하십시오.

추가 참조

문자열 데이터 구성원에 ASCII 문자를 할당 페이지의 1002

비적산 할당 지정 페이지의 1001

ST(스트럭쳐드 텍스트) 구성 요소: 식 페이지의 1003

문자열 리터럴 페이지의 1012

비적산 할당 지정

비적산 할당은 비적산 할당의 태그가 컨트롤러에서 다음을 수행할 때마다 0 으로 리셋된다는 점에서 위에서 설명한 보통 할당과 다릅니다.

- 실행 모드 진입
- 자동 리셋을 위해 SFC 를 구성한 경우 SFC 의 단계를 떠납니다. 이는 단계 동작에 할당을 포함하거나 JSR 명령어를 통해 동작을 사용하여 ST(스트럭쳐드 텍스트) 루틴을 호출하는 경우에만 적용됩니다.

비적산 할당의 구문은 다음과 같습니다.

tag [:=] expression ;

여기에서:

구성 요소	설명	
태그	새 값을 가져오는 태그를 나타내며, 태그는 BOOL, SINT, INT, DINT, STRING 또는 REAL 이어야 합니다. **팁:** STRING 태그는 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러에만 적용할 수 있습니다.	
[:=]	비적산 할당 기호입니다.	
식	태그에 할당할 새 값을 나타냅니다.	
	태그의 데이터 유형	**사용하는 식의 형식**
	BOOL	BOOL
	SINT	숫자
	INT	
	DINT	
	REAL	
	STRING (CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러만 해당).	문자열 형식, 문자열 태그 및 문자열 리터럴 포함 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380 및 GuardLogix 5580 컨트롤러(만 해당)

추가 참조

문자열 데이터 구성원에 ASCII 문자를 할당 페이지의 1002

ST(스트럭쳐드 텍스트) 구성 요소: 할당 페이지의 1000

문자열 데이터 구성원에 ASCII 문자를 할당

문자열 데이터 구성원에 ASCII 문자를 할당

할당 연산자를 사용하여 문자열 태그의 DATA 구성원의 요소에 ASCII 문자를 할당합니다. 문자를 할당하려면 문자의 값을 지정하거나 태그 이름, DATA 구성원 및 문자의 요소를 지정합니다. 예:

올바른 형식	잘못된 형식
string1.DATA[0] := 65;	string1.DATA[0] := A;
string1.DATA[0]:= string2.DATA[0];	string1 := string2; **팁:** 이 식은 한 문자 대신에 string2 의 모든 내용을 string1 에 할당합니다.

문자열 태그에 문자열을 추가하거나 삽입하려면 다음 ASCII 문자열 명령어 중 하나를 사용하십시오.

실행할 작업	사용할 명령어
문자열의 끝에 문자를 추가	CONCAT
문자열에 문자를 삽입	INSERT

추가 참조

ST(스트럭쳐드 텍스트) 구성 요소: 식 페이지의 1003

문자열 리터럴 페이지의 1012

ST(스트럭쳐드 텍스트) 구성 요소: 식

식은 태그 이름, 수식 또는 비교입니다. 식을 쓰려면 다음 중 하나를 사용합니다.

- 값을 저장하는 태그 이름(변수)
- 식에 직접 입력하는 수(직접 값)
- 식에 직접 입력하는 문자열 리터럴(CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러만 해당)
- 함수, 다음과 같음: ABS, TRUNC
- 연산자, 다음과 같음: +, -, <, >, And, Or

식을 쓸 경우 다음 지침을 따르십시오.

- 대문자와 소문자의 임의 조합을 사용합니다. 예를 들어 "AND"의 다음 변형이 허용됩니다. AND, And, and.
- 더 복잡한 요구 사항의 경우 괄호를 사용하여 식 안에서 식을 그룹화합니다. 그러면 전체 식을 읽기가 더 쉬워지며 식이 확실히 원하는 순서로 실행됩니다.

ST(스트럭쳐드 텍스트)에 다음 식을 사용할 수 있습니다.

BOOL 식: BOOL 값 1(참) 또는 0(거짓)을 생성하는 식입니다.

- 부울 식은 부울 태그, 관계 연산자 및 논리 연산자를 사용하여 값을 비교하거나 조건이 참인지 또는 거짓인지 확인합니다. 예, tag1>65.
- 간단한 부울 식은 단일 BOOL 태그일 수 있습니다.
- 일반적으로 부울 식을 사용하여 다른 로직의 실행 조건을 설정합니다.

수식: 정수 또는 부동 소수점 값을 계산하는 식입니다.

- 수식은 산술 연산자, 산술 함수 및 비트별 연산자를 사용합니다. 예, tag1+5.
- BOOL 식 내에 수식을 끼워 넣습니다. 예, (tag1+5)>65.

문자열 식: 문자열을 나타내는 식

- 간단한 식은 문자열 리터럴 또는 문자열 태그일 수 있음

이 표를 사용하여 식에 대한 연산자를 선택합니다.

다음의 경우	사용
산술 값 계산	산술 연산자 및 함수
두 값 또는 문자열 비교	관계 연산자
조건이 참인지 또는 거짓인지 확인	논리 연산자
값 내의 비트 비교	비트별 연산자

추가 참조

산술 연산자 및 함수 사용 페이지의 1005

관계 연산자 이용 페이지의 1008

논리 연산자 사용 페이지의 1007

비트별 연산자 사용 페이지의 1006

산술 연산자 및 함수 사용

산술 식에서 여러 연산자와 함수를 결합합니다.

연산자가 새 값을 계산합니다.

실행할 작업	사용 연산자	최적의 데이터 유형
더하기	+	DINT, REAL
빼기/부정	-	DINT, REAL
곱하기	*	DINT, REAL
지수(x 의 y 제곱)	**	DINT, REAL
나누기	/	DINT, REAL
모듈로-나누기	MOD	DINT, REAL

함수가 수학 연산을 수행합니다. 함수에 대해 상수, 부울이 아닌 태그 또는 식을 지정합니다.

대상	사용 함수	최적의 데이터 유형
절대값	ABS (numeric_expression)	DINT, REAL
아크 코사인	ACOS (numeric_expression)	REAL
아크 사인	ASIN (numeric_expression)	REAL
아크 탄젠트	ATAN (numeric_expression)	REAL
코사인	COS (numeric_expression)	REAL
라디안 -> 도	DEG (numeric_expression)	DINT, REAL
자연 로그	LN (numeric_expression)	REAL
로그 밑수 10	LOG (numeric_expression)	REAL
도 -> 라디안	RAD (numeric_expression)	DINT, REAL
사인	SIN (numeric_expression)	REAL
제곱근	SQRT (numeric_expression)	DINT, REAL
탄젠트	TAN (numeric_expression)	REAL
자르기	TRUNC (numeric_expression)	DINT, REAL

이 표에는 산술 연산자 및 함수 사용에 대한 예가 나와 있습니다.

사용 형식	예	
	적용 상황	쓰기
value1 operator value2	gain_4 및 gain_4_adj 가 DINT 태그이며 다음과 같은 요구가 있는 경우. 'gain_4 에 15 를 더하고 그 결과를 gain_4_adj 에 저장'"	gain_4_adj := gain_4+15;
operator value1	alarm 및 high_alarm 이 DINT 태그이며 다음과 같은 요구가 있는 경우. 'high_alarm 을 부정 값으로 하고 그 결과를 alarm 에 저장.'	alarm:= -high_alarm;
function(numeric_expression)	overtravel 및 overtravel_POS 가 DINT 태그이며 다음과 같은 요구가 있는 경우. 'overtravel 의 절대값을 계산하고 그 결과를 overtravel_POS 에 저장.'	overtravel_POS := ABS(overtravel);
value1 operator (function((value2+value3)/2)	adjustment 및 position 이 DINT 태그이고 sensor1 및 sensor2 가 REAL tag 이며 다음과 같은 요구가 있는 경우. 'sensor1 과 sensor2 의 절대값을 구하고 adjustment 를 더한 후 결과를 position 에 저장.'	position := adjustment + ABS((sensor1 + sensor2)/2);

추가 참조

ST(스트럭쳐드 텍스트) 구성 요소: 식 페이지의 1003

비트별 연산자 사용

비트별 연산자는 두 값을 기반으로 값 내의 비트를 조작합니다.

다음은 비트별 연산자의 개요를 제공합니다.

대상	사용 연산자	최적의 데이터 유형
비트별 논리곱	&, AND	DINT
비트별 논리합	또는	DINT
비트별 배타적 논리합	XOR	DINT
비트별 보수	NOT	DINT

다음은 예입니다.

사용 형식	예	
	적용 상황	사용
value1 operator value2	input1, input2 및 result1 이 DINT 태그이며 다음과 같은 요구가 있는 경우. "input1 과 input2 의 비트별 결과를 계산. 결과를 result1 에 저장."	result1 := input1 AND input2;

추가 참조

ST(스트럭쳐드 텍스트) 구성 요소: 식 페이지의 1003

논리 연산자 사용

논리 연산자를 사용하여 여러 조건이 참인지 또는 거짓인지 확인합니다. 논리 연산의 결과는 BOOL 값입니다.

비교 결과	결과
참	1
거짓	0

이 논리 연산자를 사용합니다.

실행할 비교	사용 연산자	최적의 데이터 유형
논리곱	&, AND	BOOL
논리합	또는	BOOL
배타적 논리합	XOR	BOOL
논리 보수	NOT	BOOL

이 표에는 논리 연산자 사용에 대한 예가 나와 있습니다.

사용 형식	예	
	적용 상황	사용
BOOLtag	photoeye 가 BOOL 태그이며 다음과 같은 요구가 있는 경우. "photoeye_1 이 켜지면 다음을 수행..."	IF photoeye THEN...
NOT BOOLtag	photoeye 가 BOOL 태그이며 다음과 같은 요구가 있는 경우. "If photoeye is off then..."	IF NOT photoeye THEN...

expression1 & expression2	photoeye 가 BOOL 태그이면 temp 는 DINT 태그이며 다음과 같은 요구가 있는 경우. "photoeye 가 켜지고 temp 가 100 보다 작으면 다음을 수행..."	IF photoeye & (temp<100) THEN...
expression1 OR expression2	photoeye 가 BOOL 태그이면 temp 는 DINT 태그이며 다음과 같은 요구가 있는 경우. "photoeye 가 켜지거나 temp 가 100 보다 작으면 다음을 수행...".	IF photoeye OR (temp<100) THEN...
expression1 XOR expression2	photoeye1 과 photoeye2 가 BOOL 태그이며 다음과 같은 요구가 있는 경우. "다음의 경우: photoeye2 가 꺼진 동안 photoeye1 이 켜지거나 또는 photoeye2 가 켜진 동안 photoeye1 이 꺼지면 다음을 수행..."	IF photoeye1 XOR photoeye2 THEN...
BOOLtag := expression1 & expression2	photoeye1 과 photoeye2 가 BOOL 태그이고 open 은 BOOL 태그이며 다음과 같은 요구가 있는 경우. "photoeye1 과 photoeye2 가 둘 다 켜지면 open 을 참으로 설정"	open := photoeye1 & photoeye2;

추가 참조

ST(스트럭쳐드 텍스트) 구성 요소: 식 페이지의 1003

관계 연산자 이용

관계 연산자는 두 값 또는 문자열을 비교하여 참 또는 거짓 결과를 제공합니다. 관계 연산의 결과는 BOOL 값입니다.

비교 결과	결과
참	1
거짓	0

이 관계 연산자를 사용합니다.

실행할 비교	사용 연산자	최적의 데이터 유형
같음	=	DINT, REAL, 문자열 형식
보다 작음	<	DINT, REAL, 문자열 형식
보다 작거나 같음	<=	DINT, REAL, 문자열 형식
보다 큼	>	DINT, REAL, 문자열 형식
보다 크거나 같음	>=	DINT, REAL, 문자열 형식
같지 않음	<>	DINT, REAL, 문자열 형식

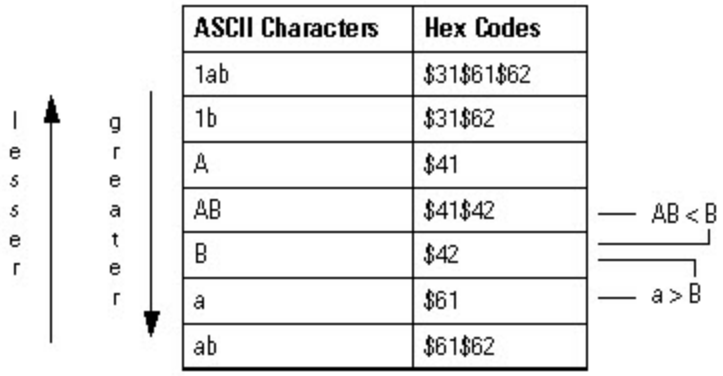
이 표에는 관계 연산자 사용에 대한 예가 나와 있습니다.

사용 형식	예	
	적용 상황	쓰기
value1 operator value2	temp 가 DINT 태그이며 다음과 같은 요구가 있는 경우. 'temp 가 100 보다 작으면 다음을 수행…'	IF temp<100 THEN...
stringtag1 operator stringtag2	bar_code 및 dest 가 문자열 태그이며 다음과 같은 요구가 있는 경우. 'bar_code 가 dest 와 같으면 다음을 수행…'	IF bar_code=dest THEN...
stringtag1 operator 'character string literal'	bar_code 가 문자열 태그이며 다음과 같은 요구가 있는 경우. 'bar_code 가 'Test PASSED'와 같으면 다음을 수행...'	IF bar_code='Test PASSED' THEN...
char1 operator char2 식에 ASCII 문자를 직접 입력하려면 해당 문자의 십진 값을 입력합니다.	bar_code 가 문자열 태그이며 다음과 같은 요구가 있는 경우. 'bar_code.DATA[0]이 'A'와 같으면 다음을 수행…'	IF bar_code.DATA[0]=65 THEN...
bool_tag := bool_expressions	count 와 length 가 DINT 태그이고 done 이 BOOL 태그이며 다음과 같은 요구가 있는 경우. 'count 가 length 보다 크거나 같으면 카운트를 수행.'	Done := (count >= length);

문자열을 계산하는 방법

ASCII 문자의 16 진 값은 한 문자열이 다른 문자열보다 작은지 아니면 큰지를 결정합니다.

- 두 문자열이 전화 번호부에서와 같이 정렬되는 경우, 문자열의 순서가 더 큰 것이 어느 것인지 결정합니다.



- 문자들이 일치하면 문자열이 같습니다.

- 문자는 대/소문자를 구분합니다. 대문자 "A"($41)는 소문자 "a"($61)와 같지 않습니다.

추가 참조

ST(스트럭쳐드 텍스트) 구성 요소: 식 페이지의 1003

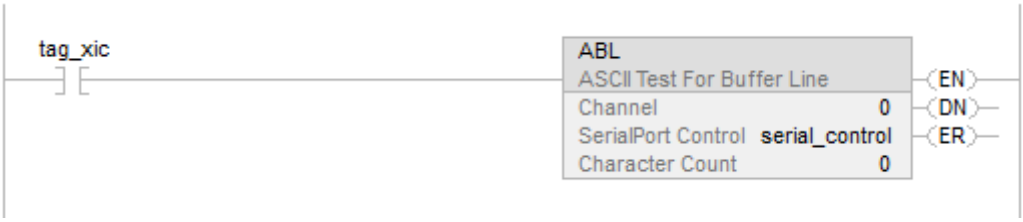
ST(스트럭쳐드 텍스트) 구성 요소: 명령어

ST(스트럭쳐드 텍스트) 명령문은 명령어일 수도 있습니다. ST(스트럭쳐드 텍스트) 명령어는 스캔될 때마다 실행됩니다. 구성 내의 ST(스트럭쳐드 텍스트) 명령어는 구성의 조건이 참일 때마다 실행합니다. 구성의 조건이 거짓이면 구성 내의 명령문이 스캔되지 않습니다. 실행을 트리거하는 렁-조건 또는 상태 전환은 없습니다.

이는 EnableIn 을 사용하여 실행을 트리거하는 평션 블록 명령어와 다릅니다. ST(스트럭쳐드 텍스트) 명령어는 EnableIn 이 항상 설정된 것처럼 실행합니다.

이는 렁-입력-조건을 사용하여 실행을 트리거하는 래더 다이어그램 명령어와도 다릅니다. 일부 래더 다이어그램 명령어는 렁-입력-조건이 거짓에서 참으로 전환할 때만 실행합니다. 이는 전환적인 래더 다이어그램 명령어입니다. ST(스트럭쳐드 텍스트)에서 명령어는 ST(스트럭쳐드 텍스트) 명령어 실행의 사전 조건을 설정하지 않은 한 스캔될 때 실행합니다.

예를 들어 ABL 명령어는 래더 다이어그램의 전환 명령어입니다. 이 예에서 ABL 명령어는 tag_xic 가 해제됨에서 설정됨으로 전환할 때 스캔 시에만 실행합니다. ABL 명령어는 tag_xic 가 설정된 상태로 유지되거나 tag_xic 가 해제될 때 실행하지 않습니다.



ST(스트럭쳐드 텍스트)에서 이 예를 다음과 같이 쓴다면:

```
IF tag_xic THEN ABL(0,serial_control);

END_IF;
```

ABL 명령어는 tag_xic 가 해제됨에서 설정됨으로 전환할 때뿐만 아니라 tag_xic 가 설정된 모든 스캔도 실행합니다.

ABL 명령어가 tag_xic 가 해제됨에서 설정됨으로 전환할 때만 실행하도록 하려면 ST(스트럭쳐드 텍스트) 명령어의 조건을 설정해야 합니다. 원샷을 사용하여 실행을 트리거합니다.

```
osri_1.InputBit := tag_xic;

OSRI(osri_1);


IF (osri_1.OutputBit) THEN

ABL(0,serial_control);

END_IF;
```

ST(스트럭쳐드 텍스트) 구성 요소: 구성

구조를 단독으로 프로그래밍하거나 다른 구조 내에 중첩합니다.

다음의 경우	구성:
특정 조건이 발생하는 경우 실행	IF. . . THEN
숫자 값을 기반으로 할 일 선택	CASE. . . OF
다른 일을 수행하기 전에 무언가를 특정 횟수만큼 수행	FOR. . . DO
특정 조건이 참인 경우 무언가를 계속 수행	WHILE. . . DO
조건이 참이 될 때까지 무언가를 계속 수행	REPEAT. . . UNTIL

일부 키워드는 예약되어 있음

다음 구성은 사용할 수 없음:

- GOTO
- REPEAT

Logix Designer 응용 프로그램을 사용하여 이들을 태그 이름 또는 구성으로 사용할 수 없습니다.

추가 참조

IF_THEN 페이지의 1020

CASE_OF 페이지의 1014

FOR_DO 페이지의 1016

WHILE_DO 페이지의 1026

REPEAT_UNTIL 페이지의 1023

문자열 리터럴

문자열 리터럴은 1 바이트 또는 2 바이트 인코딩된 문자를 포함합니다. 1 바이트 문자열 리터럴은 작은 따옴표 문자(')로 시작하고 작은 따옴표 문자로 끝나는 문자 0 개 이상의 시퀀스입니다. 1 바이트 문자열에서 달러 부호($)와 그 뒤의 16 진수 2 개로 이루어진 3 문자 조합은 다음 표에 나오는 8 비트 문자 코드의 16 진수 표시로 해석됩니다.

팁:

- 문자열 리터럴은 CompactLogix 5380, CompactLogix 5480, ControlLogix 5580, Compact GuardLogix 5380, GuardLogix 5580 컨트롤러에만 적용할 수 있습니다.
- Studio 5000 는 1 바이트 문자만 지원합니다.

문자열 리터럴

아니요.	설명	예
1a	빈 문자열(길이 0)	''
1b	단일 문자를 포함하는 길이 1 또는 문자 CHAR 의 문자열	'A'
1c	"공백" 문자를 포함하는 길이 1 또는 문자 CHAR 의 문자열	' '
1d	"작은 따옴표" 문자를 포함하는 길이 1 또는 문자 CHAR 의 문자열	'$''
1e	"큰 따옴표" 문자를 포함하는 길이 1 또는 문자 CHAR 의 문자열	'"'
1f	두 문자 조합을 지원	'RL'
1g	'$'와 16 진 문자 2 개로 이루어진 문자 표시 지원	'$0A'

문자 문자열의 2 문자 조합

아니요.	설명	예
1	달러 부호	$$
2	작은 따옴표	$'
3	개행	$L 또는 $l
4	새 줄	$N 또는 $n
5	용지 먹임(페이지)	$P 또는 $p
6	캐리지 리턴	$R 또는 $r
7	도표 작성기	$T 또는 $t

팁:
- 새 줄 문자는 물리적 I/O 와 파일 I/O 둘 다에 대해 데이터 줄의 끝을 정의하는 구현 독립적인 수단을 제공합니다. 인쇄의 경우 이렇게 한 효과는 데이터 줄을 끝내고 다음 줄의 시작에서 인쇄를 다시 시작하는 것입니다.
- $' 조합은 작은 따옴표로 둘러싼 문자열 리터럴 내에서만 유효합니다.

추가 참조

ST(스트럭쳐드 텍스트) 구성 요소: 할당 페이지의 1000

문자열 형식 페이지의 904

문자열 형식

태그에 문자열 형식 데이터 유형을 사용하는 ASCII 문자가 저장됩니다.

- 기본값 STRING 데이터 유형을 사용하여 최대 82 문자를 저장
- 더 적거나 많은 문자를 저장하는 새 문자열 형식을 만듦

새 문자열 형식을 작성하려면 LOGIX 5000 Controllers ASCII Strings Programming Manual (발행 번호 1756-PM013)을 참조하십시오.

각 문자열 형식은 다음 구성원을 포함합니다.

이름(Name)	데이터 유형	설명	참고
LEN	DINT	문자열의 문자 수	다음을 사용할 때마다 LEN 이 새 문자 카운트로 자동 업데이트됩니다. • 문자 입력을 위한 문자열 브라우저 • 문자열을 읽거나, 변환하거나 조작하는 명령어 LEN 은 현재 문자열의 길이를 표시합니다. DATA 구성원은 LEN 카운트에 포함되지 않는 추가적인 이전 문자를 포함할 수 있습니다.
DATA	SINT 배열	문자열의 ASCII 문자	문자열의 문자에 액세스하려면 태그의 이름에 따라 주소를 검색합니다. 예를 들어 string_1 태그의 문자에 액세스하려면 string_1 을 입력합니다. DATA 배열의 각 요소는 한 문자를 포함합니다. 그보다 적거나 많은 문자를 저장하는 새로운 문자열 형식을 작성하십시오.

추가 참조

문자열 리터럴 페이지의 1012

CASE_OF

CASE_OF 를 사용하여 숫자 값을 기반으로 수행할 작업을 선택합니다.

피연산자

CASE numeric_expression OF

selector1: statement;

selectorN: statement; ELSE

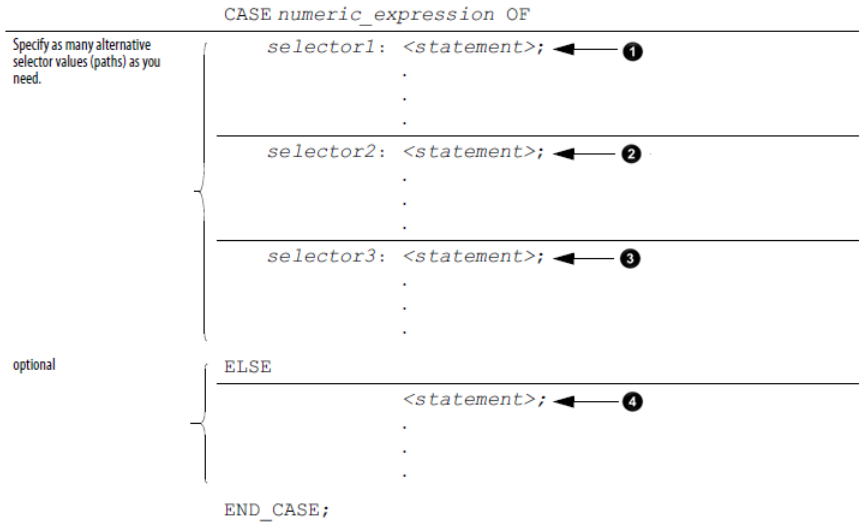
ST(스트럭쳐드 텍스트)

피연산자	유형	형식	입력
Numeric_ expression	SINT INT DINT REAL	태그 식	숫자로 계산되는 태그 또는 식 (수식)
Selector	SINT INT DINT REAL	즉시	numeric_expression 과 같은 유형

중요:	REAL 값을 사용하는 경우, REAL 값은 하나의 특정 값을 정확히 일치시키는 것보다는 값의 범위 내에 있을 가능성이 더 높기 때문에 셀렉터에 대해 일정 범위의 값을 사용십시오.

설명

구문은 표에 설명되었습니다.



이들은 셀렉터 값을 입력하기 위한 구문입니다.

셀렉터가 다음과 같은 경우	입력
한 값	value: statement
여러 개의 구분되는 값	value1, value2, valueN : <statement> 각 값을 구분하려면 쉼표(,)를 사용합니다.
일정 범위의 값	value1..valueN : <statement> 범위를 식별하려면 마침표 2 개(..)를 사용합니다.
구분되는 값 + 일정 범위의 값	valuea, valueb, value1..valueN : <statement>

CASE 구성은 C 또는 C++ 프로그래밍 언어의 스위치 명령문과 유사합니다. CASE 구성에서는 컨트롤러가 첫 번째 일치하는 셀렉터 값과 연결된 문만 실행합니다. 실행은 해당 셀렉터의 명령문 뒤에서 항상 분리되고 END_CASE 명령문으로 이동합니다.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

없음

예

원하는 작업	입력할 ST(스트럭쳐드 텍스트)
레시피 번호 = 1 이면 재료 A 출구 1 = 개방(1) 재료 B 출구 4 = 개방(1)	CASE recipe_number OF
	1: Ingredient_A.Outlet_1 :=1; Ingredient_B.Outlet_4 :=1;
레시피 번호 = 2 또는 3 이면 재료 A 출구 4 = 개방(1) 재료 B 출구 2 = 개방(1)	2,3: Ingredient_A.Outlet_4 :=1; Ingredient_B.Outlet_2 :=1;
레시피 번호 = 4, 5, 6 또는 7 이면 재료 A 출구 4 = 개방(1) 재료 B 출구 2 = 개방(1)	4 ~ 7: Ingredient_A.Outlet_4 :=1; Ingredient_B.Outlet_2 :=1;
레시피 번호 = 8, 11, 12 또는 13 이면 재료 A 출구 1 = 개방(1) 재료 B 출구 4 = 개방(1)	8,11…13 Ingredient_A.Outlet_1 :=1; Ingredient_B.Outlet_4 :=1;
그렇지 않으면 모든 출구 = 폐쇄(0)	ELSE
	Ingredient_A.Outlet_1 [:=]0; Ingredient_A.Outlet_4 [:=]0; Ingredient_B.Outlet_2 [:=]0; Ingredient_B.Outlet_4 [:=]0;
	END_CASE;

[:=] 는 컨트롤러에 대해 컨트롤러 태그가 다음을 수행할 때마다 출구 태그도 해제할 것을 지정함:

RUN 모드로 진입합니다.

SFC 를 자동 리셋으로 구성한 경우 SFC 의 단계를 떠납니다. 이는 단계 동작에 할당을 포함하거나 JSR 명령어를 통해 동작을 사용하여 ST(스트럭쳐드 텍스트) 루틴을 호출하는 경우에만 적용됩니다.

FOR_DO

FOR_DO 루프를 사용하여 작업을 특정 횟수 수행한 후에 다른 작업을 수행합니다.

활성화된 경우 FOR 명령어는 Index 가 Terminal value 를 초과할 때까지 루틴을 반복적으로 실행합니다. 단계 값은 양수 또는 음수일 수 있습니다. 음수일 경우 인덱스가 터미널 값보다 작으면

루프가 종료됩니다. 양수일 경우 인덱스가 터미널 값보다 클 때 루프가 종료됩니다.

FOR 명령어가 루틴을 실행할 때마다 인덱스에 단계 크기가 추가됩니다.

단일 스캔에서 너무 많이 루프하지 마십시오. 너무 많이 반복할 경우 컨트롤러의 워치독이 타임아웃되어 메이저 폴트가 발생할 수 있습니다.

피연산자

FOR count:= initial_value TO

final_value BY increment DO

<statement>;

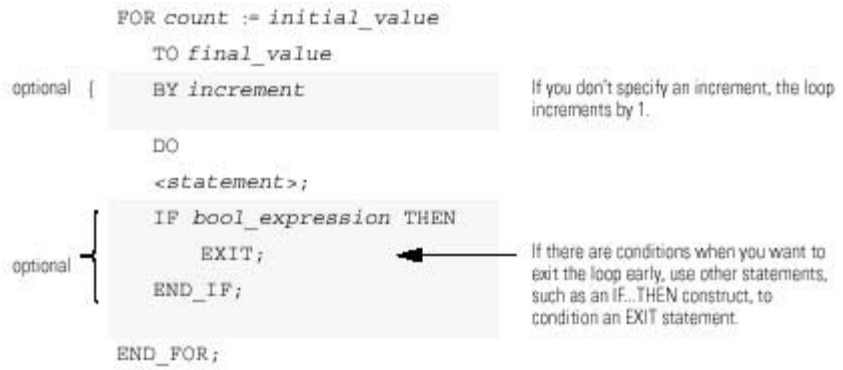
END_FOR;

피연산자	유형	형식	설명
count	SINT INT DINT	태그	FOR_DO 가 실행될 때 카운트 위치를 저장하는 태그
initial_ value	SINT INT DINT	택 식 즉시	숫자로 계산되어야 함 카운트에 대한 초기값 지정
final_ value	SINT INT DINT	택 식 즉시	루프를 종료하는 시기를 결정하는 카운트에 대한 최종 값을 지정
increment	SINT INT DINT	택 식 즉시	(옵션)루프를 통해 매번 카운트를 증가시키는 양 증가량을 지정하지 않으면 카운트는 1 단위로 증가합니다.

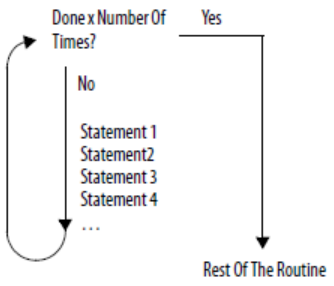
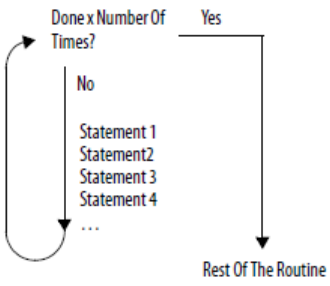
중요: 단일 스캔에 루프 내에서 너무 많이 반복하지 마십시오.
컨트롤러는 루프를 완료한 후에야 루틴 내의 다른 문을 실행합니다.
루프 완료 시간이 태스크의 워치독 타이머 시간보다 오래 걸릴 경우 메이저 폴트가 발생합니다.
IF_THEN 같은 다른 구성을 사용해 보십시오.

설명

구문은 표에 설명되었습니다.



이 다이어그램에는 FOR_DO 루프의 실행 방법과 EXIT 문이 일찍 루프에서 떠나는 방법이 나와 있습니다.

	
FOR_DO 루프는 특정 횟수만큼 실행됩니다.	카운트가 마지막 값에 도달하기 전에 루프를 중지하려면 EXIT 문을 사용하십시오.

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

메이저 폴트가 발생하는 원인	폴트 유형	폴트 코드
구성 루프가 너무 깁니다.	6	1

예 1

다음을 수행할 경우	입력할 ST(스트럭쳐드 텍스트)

BOOL 배열에서 0 ~ 31 비트 해제:
subscript 태그를 0 으로 초기화합니다.
i 를 지웁니다. 예를 들어 subscript = 5 일 경우 배열[5]를 지웁니다.
subscript 에 1 을 더합니다.
subscript ≤ 31 일 경우 2 및 3 을 반복합니다.
아니면 중지합니다.
	For subscript:=0 to 31 by 1 do
	array[subscript] := 0;
	End_for;

예 2

다음을 수행할 경우	입력할 ST(스트럭쳐드 텍스트)

사용자 정의 데이터 유형(구조)에는 인벤토리 내 항목에 대한 다음 정보가 저장됩니다.
• 항목의 바코드 ID(문자열 데이터 유형)
• 항목의 재고 수량(DINT 데이터 유형)
위 구조 배열에 인벤토리 내 다양한 항목별 요소가 포함되어 있습니다. 특정 제품의 배열을 검색하고(제품 바코드 사용) 재고 수량을 파악하려고 합니다.
1. 인벤토리 배열의 크기(항목 수)를 가져와서 결과를
2. Inventory_Items(DINT 태그)에 저장합니다.
위치 태그를 0 으로 초기화합니다.
3. Barcode 가 배열 내 항목 ID 와 일치하면:
Quantity 태그 = Inventory[position].Qty 로 설정합니다. 이렇게 하면 항목의 재고 수량이 산출됩니다.
정지합니다.
Barcode 는 검색하려는 항목의 바코드가 저장된 문자열 태그입니다. 예를 들어
position = 5 일 경우 Barcode 를 Inventory[5].ID.와 비교합니다.
4. position 에 1 을 더합니다.
5. position ≤ to (Inventory_Items -1)일 경우 3 및 4 를 반복합니다. 요소 번호가 0 부터 시작하므로 마지막 요소는 배열의 요소 수보다 1 만큼 작습니다.
아니면 중지합니다.
	SIZE(Inventory,0,Inventory_Items);
	For position:=0 to Inventory_Items - 1 do
	If Barcode = Inventory[position].ID then
	Quantity := Inventory[position].Qty;
	Exit;
	End_if;
	End_for;

IF_THEN

특정 조건이 발생할 경우 IF_THEN 을 사용하여 작업을 완료합니다.

피연산자

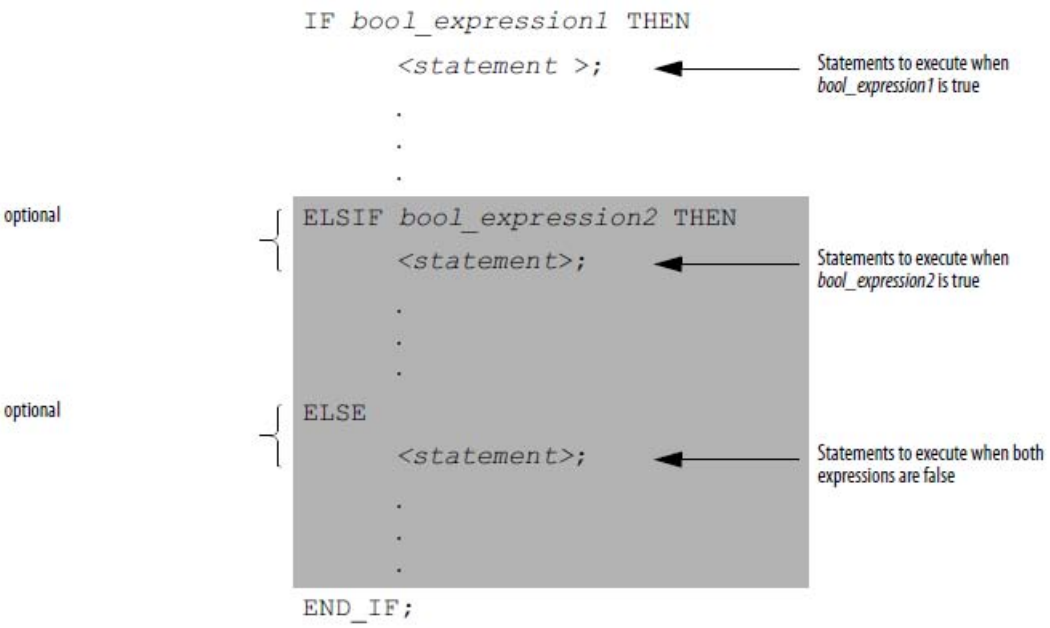
IF bool_expression THEN

<statement>;

피연산자	유형	형식	입력
Bool_ expression	BOOL	태그 식	BOOL 값으로 계산되는 BOOL 태그 또는 식 (BOOL 식)

설명

구문은 표에 설명되었습니다.



ELSIF 또는 ELSE 구문을 사용하려면 다음 가이드라인을 따르십시오.

사용 가능한 몇 개의 문 그룹에서 선택하려면 하나 이상의 ELSIF 문을 추가합니다.

ELSIF 문마다 하나의 대체 경로를 나타냅니다.

필요한 수의 ELSIF 경로를 지정하십시오.

컨트롤러는 첫 번째 참인 IF 또는 ELSIF 문을 실행하고 나머지 ELSIF 및 ELSE 문을 건너뜁니다.

모든 IF 또는 ELSIF 조건이 거짓일 경우 어떤 작업이 실행되도록 하려면 ELSE 문을 추가하십시오.

다음 표에 IF, THEN, ELSIF 및 ELSE 문의 다양한 조합이 요약되어 있습니다.

다음의 경우	및	구성:
조건이 참일 경우 작업 실행	조건이 거짓일 경우 아무 작업도 실행하지 않음	IF_THEN
	조건이 거짓일 경우 다른 작업 실행	IF_THEN_ELSE
입력 조건에 따라 대체 문 또는 문 그룹 선택	조건이 거짓일 경우 아무 작업도 실행하지 않음	IF_THEN_ELSIF
	모든 조건이 거짓일 경우 기본 문 할당	IF_THEN_ELSIF_ELSE

연산 상태 플래그에 영향

아니요

메이저/마이너 폴트

없음.

예제

예 1

IF...THEN

이 작업을 수행할 경우	입력할 ST(스트럭쳐드 텍스트)
IF rejects > 3 then	IF rejects > 3 THEN
conveyor = off (0)	conveyor := 0;
alarm = on (1)	alarm := 1;
	END_IF;

예 2

IF_THEN_ELSE

이 작업을 수행할 경우	입력할 ST(스트럭쳐드 텍스트)
If conveyor direction contact = forward (1) then	IF conveyor_direction THEN
light = off	light := 0;
Otherwise light = on	ELSE
	light [:=] 1;
	END_IF;

[:=] 기호는 컨트롤러가 다음을 수행할 때마다 라이트 커튼을 해제하도록 지시합니다.

RUN 모드로 진입합니다.

자동 리셋을 위해 SFC 를 구성한 경우 SFC 의 단계를 떠납니다. (이는 단계 동작에 할당을 포함하거나 JSR 명령어를 통해 동작을 사용하여 ST(스트럭쳐드 텍스트) 루틴을 호출하는 경우에만 적용됩니다.)

예 3

IF...THEN...ELSIF

이 작업을 수행할 경우	입력할 ST(스트럭쳐드 텍스트)
If sugar low limit switch = low (on) and sugar high limit switch = not high (on) then	IF Sugar.Low & Sugar.High THEN
inlet valve = open (on)	Sugar.Inlet [:=] 1;
Until sugar high limit switch = high (off)	ELSIF NOT(Sugar.High) THEN
	Sugar.Inlet := 0;
	END_IF;

[:=] 기호는 컨트롤러가 다음을 수행할 때마다 Sugar.Inlet 을 해제하도록 지시합니다.

RUN 모드로 진입합니다.

자동 리셋을 위해 SFC 를 구성한 경우 SFC 의 단계를 떠납니다. (이는 단계 동작에 할당을 포함하거나 JSR 명령어를 통해 동작을 사용하여 ST(스트럭쳐드 텍스트) 루틴을 호출하는 경우에만 적용됩니다.)

예 4

IF...THEN...ELSIF...ELSE

이 작업을 수행할 경우	입력할 ST(스트럭쳐드 텍스트)
If tank temperature > 100	IF tank.temp > 200 THEN
then pump = slow	pump.fast :=1; pump.slow :=0; pump.off :=0;
If tank temperature > 200	ELSIF tank.temp > 100 THEN
then pump = fast	pump.fast :=0; pump.slow :=1; pump.off :=0;
Otherwise pump = off	ELSE
	pump.fast :=0; pump.slow :=0; pump.off :=1;
	END_IF;

REPEAT_UNTIL

조건이 참일 때까지 REPEAT_UNTIL 루프를 사용하여 작업을 계속 수행합니다.

피연산자

REPEAT

<statement>;

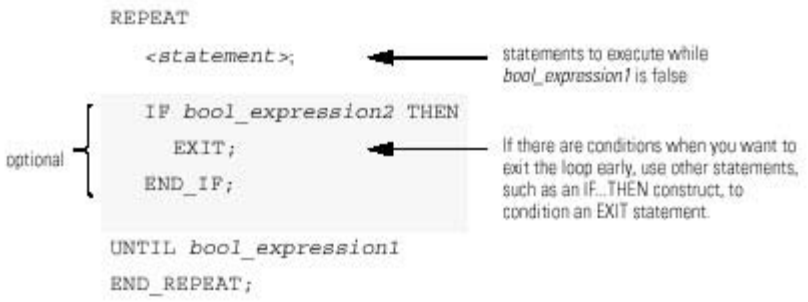
ST(스트럭쳐드 텍스트)

피연산자	유형	형식	입력
bool_ expression	BOOL	태그 식	BOOL 값으로 계산되는 BOOL 태그 또는 식 (BOOL 식)

중요:
- 단일 스캔에 루프 내에서 너무 많이 반복하지 마십시오.
- 컨트롤러는 루프를 완료한 후에야 루틴 내의 다른 문을 실행합니다.
- 루프 완료 시간이 태스크의 워치독 타이머 시간보다 오래 걸릴 경우 메이저 폴트가 발생합니다.
- IF_THEN 같은 다른 구성을 사용해 보십시오.

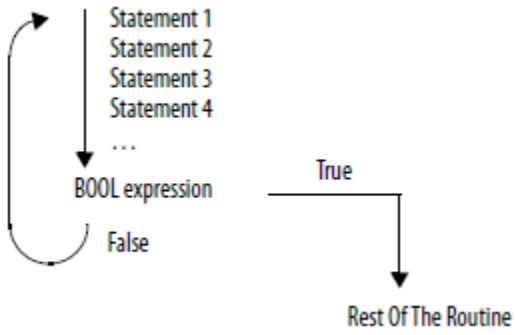
설명

구문:

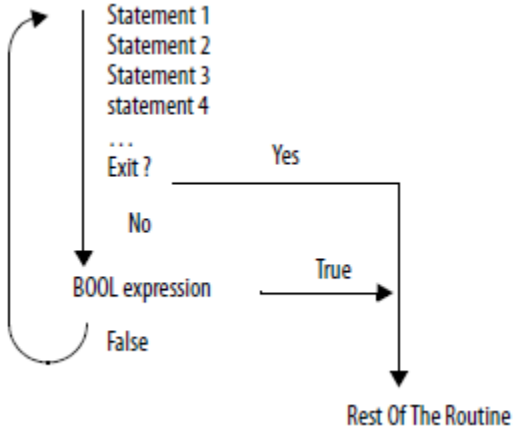


다음 다이어그램에는 REPEAT_UNTIL 루프의 실행 방법과 EXIT 문이 일찍 루프에서 떠나는 방법이 나와 있습니다.

bool_expression 이 거짓인 동안 컨트롤러는 REPEAT_UNTIL 루프 내의 문만 실행합니다.



조건이 거짓으로 되기 전에 루프를 중지하려면 EXIT 문을 사용하십시오.



연산 상태 플래그에 영향

아니요

폴트 조건

메이저 폴트가 발생하는 원인	폴트 유형	폴트 코드
구성 루프가 너무 깁니다.	6	1

예 1

다음을 수행할 경우	입력할 ST(스트럭쳐드 텍스트)

REPEAT_UNTIL 루프의 경우 구성 내의 문을 실행한 다음 조건이 참인지 판단한 후에 다시 문을 실행합니다. 이는 WHILE_DO 루프와 다릅니다. WHILE_DO 루프는 조건을 먼저 평가하기 때문입니다.
조건이 참이면 컨트롤러가 루프 내의 문을 실행합니다.
REPEAT_UNTIL 루프의 문은 항상 최소 1 회 실행됩니다. WHILE_DO 루프의 문은 한 번도 실행되지 않을 수도 있습니다.
	pos := -1;
	REPEAT
	pos := pos + 2;
	UNTIL ((pos = 101) OR (structarray[pos].value = targetvalue))
	end_repeat;

예 2

다음을 수행할 경우	입력할 ST(스트럭쳐드 텍스트)

SINT 배열의 ASCII 문자를 문자열 태그로 이동합니다. (SINT 배열의 각 요소별로 하나의 문자가 할당되어 있습니다.) 캐리지 리턴에 이르면 중지됩니다.
Element_number 를 0 으로 초기화합니다.
SINT_array(ASCII 문자가 들어 있는 배열)의 요소 수를 카운트하고 결과 값은 SINT_array_size(DINT 태그)에 저장합니다.
String_tag[element_number] = SINT_array[element_number]의 문자로 설정합니다.
element_number 에 1 을 더합니다. 이러면 컨트롤러가 SINT_array 의 다음 문자를
확인합니다.
String_tag 의 길이 구성원 = element_number 로 설정합니다. (이러면 지금까지 String_tag 의 문자 수가 기록됩니다.)
element_number = SINT_array_size 이면 중지됩니다. (배열의 끝에 있으며 캐리지 리턴이 포함되어 있지 않습니다.)
SINT_array[element_number] 문자 수 = 13(캐리지 리턴 십진수 값)일 경우 중지합니다.
	element_number := 0;
	SIZE(SINT_array, 0, SINT_array_size);
	Repeat
	String_tag.DATA[element_number] := SINT_array[element_number];
	element_number := element_number + 1;
	String_tag.LEN := element_number;
	If element_number = SINT_array_size then
	exit;
	end_if;
	Until SINT_array[element_number] = 13
	end_repeat;

WHILE_DO

어떤 조건이 참인 한 WHILE_DO 루프를 사용하여 작업을 계속 수행합니다.

피연산자

WHILE bool_expression DO

<statement>;

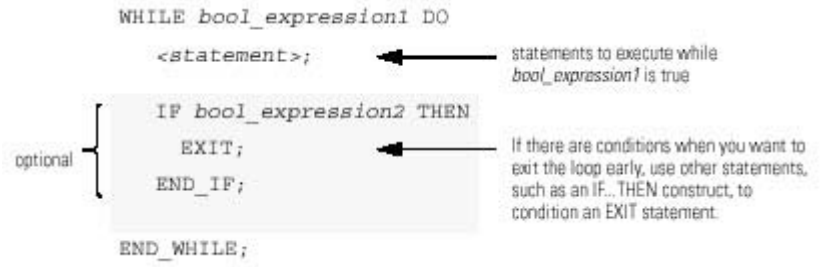
ST(스트럭쳐드 텍스트)

피연산자	유형	형식	설명
bool_expression	BOOL	태그 expression	BOOL 값으로 계산되는 BOOL 태그 또는 식

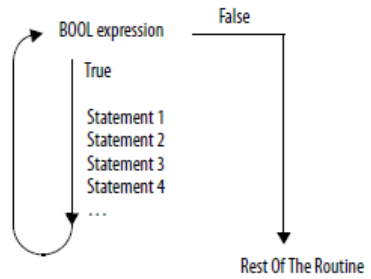
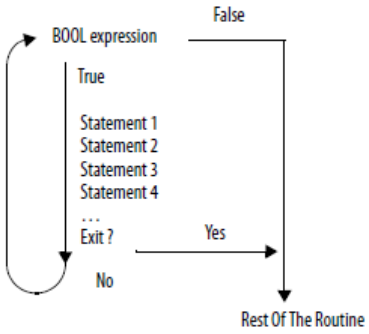
중요:	단일 스캔에 루프 내에서 너무 많이 반복하지 마십시오. 컨트롤러는 루프를 완료한 후에야 루틴 내의 다른 문을 실행합니다. 루프 완료 시간이 태스크의 워치독 타이머 시간보다 오래 걸릴 경우 메이저 폴트가 발생합니다. IF_THEN 같은 다른 구성을 사용해 보십시오.

설명

구문:



다음 다이어그램에는 WHILE_DO 루프의 실행 방법과 EXIT 문이 일찍 루프에서 떠나는 방법이 나와 있습니다.

	
bool_expression 이 참일 때 컨트롤러는 WHILE_DO 루프 내의 문만을 실행합니다.	조건이 참으로 되기 전에 루프를 중지하려면 EXIT 문을 사용하십시오.

연산 상태 플래그에 영향

아니요

폴트 조건

메이저 폴트가 발생하는 원인	폴트 유형	폴트 코드
구성 루프가 너무 깁니다.	6	1

예 1

다음을 수행할 경우	입력할 ST(스트럭쳐드 텍스트)

WHILE_DO 루프에서 그 조건을 먼저 평가합니다. 조건이 참이면 컨트롤러가 루프 내의 문을 실행합니다.
REPEAT_UNTIL 루프는 이와 다릅니다. REPEAT_UNTIL 의 경우 구성 내의 문을 실행한 다음 조건이 참인지 판단한 후에 다시 문을 실행하기 때문입니다.. REPEAT_UNTIL 루프의 문은 항상 최소 1 회 실행됩니다. WHILE_DO 루프의 문은 한 번도 실행되지 않을 수도 있습니다.
	pos := 0;
	While ((pos <= 100) & structarray[pos].value <> targetvalue)) do
	  pos := pos + 2;
	  String_tag.DATA[pos] := SINT_array[pos];
	end_while;

예 2

다음을 수행할 경우	입력할 ST(스트럭쳐드 텍스트)

SINT 배열의 ASCII 문자를 문자열 태그로 이동합니다. (SINT 배열의 각 요소별로 하나의 문자가 할당되어 있습니다.) 캐리지 리턴에 이르면 중지됩니다.
Element_number 를 0 으로 초기화합니다.
SINT_array(ASCII 문자가 들어 있는 배열)의 요소 수를
	element_number := 0;
	SIZE(SINT_array, 0, SINT_array_size);
	While SINT_array[element_number] <> 13 do
	String_tag.DATA[element_number] := SINT_array[element_number];
	element_number := element_number + 1;

카운트하고 결과 값은 SINT_array_size(DINT 태그)에 저장합니다. SINT_array[element_number] 문자 수 = 13(캐리지 리턴 십진수 값)일 경우 중지합니다. String_tag[element_number] = SINT_array[element_number] 문자로 설정합니다. element_number 에 1 을 더합니다. 이러면 컨트롤러가 SINT_array 의 다음 문자를 확인합니다. String_tag 의 길이 구성원 = element_number 로 설정합니다. (이러면 지금까지 String_tag 의 문자 수가 기록됩니다.) element_number = SINT_array_size 이면 중지됩니다. (배열의 끝에 있으며 캐리지 리턴이 포함되어 있지 않습니다.)	String_tag.LEN := element_number; If element_number = SINT_array_size then exit; end_if; end_while;

ST(스트럭쳐드 텍스트) 속성

ST(스트럭쳐드 텍스트) 프로그래밍에 고유한 문제에 관한 자세한 내용을 보려면 아래 항목을 클릭하십시오. ST(스트럭쳐드 텍스트) 프로그래밍이 실행되는 방법을 이해하려면 이 정보를 검토합니다.

추가 참조

ST(스트럭쳐드 텍스트) 구성 요소: 할당 페이지의 1000

ST(스트럭쳐드 텍스트) 구성 요소: 식 페이지의 1003

ST(스트럭쳐드 텍스트) 명령어 페이지의 1010

ST(스트럭쳐드 텍스트) 구성 요소: 구성 페이지의 1011

ST(스트럭쳐드 텍스트) 구성 요소: 주석 페이지의 999

A

B

C

D

E

F

G

I

J

L

M

N

O

P

R

S

T

U

W

X

같

계

비

빼

사

상

서

소

숫

시

실

아

알

에

열

외

타

탄

특

파

하

Rockwell Automation 지원

Rockwell Automation 은 웹에서 제품을 사용하는 데 도움이 되는 기술 정보를 제공합니다. http://www.rockwellautomation.com/support에서 기술 및 응용 프로그램 참고사항, 샘플 코드 및 소프트웨어 서비스 팩에 대한 링크를 찾을 수 있습니다. https://rockwellautomation.custhelp.com의 당사 지원 센터를 방문하여 소프트웨어 업데이트, 지원 채팅 및 포럼, 기술 정보, FAQ 를 찾아보고 제품 알림 업데이트를 위한 등록을 할 수 있습니다.

또한 당사는 설치, 구성 및 문제해결에 대한 여러 지원 프로그램을 제공합니다. 자세한 내용은 현지 대리점 또는 Rockwell Automation 담당자에게 문의하거나 http://www.rockwellautomation.com/services/online-phone을 방문하십시오.

설치 지원

설치한 후 처음 24 시간 이내에 문제가 생긴 경우 이 설명서에 포함된 정보를 확인하십시오. 제품 가동 및 실행에 있어 도움이 필요한 경우 고객 지원 센터에 문의할 수 있습니다.

미국 또는 캐나다	1.440.646.3434
미국 또는 캐나다 외부	http://www.rockwellautomation.com/locations에서 사용할 수 있는 Worldwide Locator 를 사용하거나 현지 Rockwell Automation 담당자에게 문의하십시오.

새 제품 반품 만족도

Rockwell Automation 은 제조 시설에서 배송할 때 모든 제품을 테스트하여 완전하게 작동 가능한지 확인합니다. 그러나 제품이 작동하지 않아 반품해야 하는 경우 아래 절차를 따르십시오.

미국	대리점에 문의하십시오. 반품 프로세스를 완료하려면 고객 지원 케이스 번호(번호를 받으려면 앞의 전화 번호로 전화하십시오)를 대리점에 제공해야 합니다.
미국 외부	반품 절차는 현지 Rockwell Automation 담장자에게 문의하십시오.

문서 피드백

고객의 의견은 당사가 고객의 문서 요구에 더 잘 부응하는 데 도움이 됩니다. 이 문서를 개선하는 방법에 관한 제안이 있는 경우 피드백 양식(발행 번호 RA-DU002)를 작성해 주십시오.

Rockwell Otomasyon Ticaret A.Ş., Kar Plaza İş Merkezi E Blok Kat:6 34752 İçerenköy, İstanbul, Tel: +90 (216) 5698400

www.rockwellautomation.com

Power, Control and Information Solutions Headquarters

Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444
Europe/Middle East/Africa: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640
Asia Pacific: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

본 사: 서울특별시 강남구 논현로 430 아세아타워 6층, 7층 (135-719) Tel: 02-2188-4400 **www.rockwellautomation.com/ko_KR**
부산지사: 부산광역시 해운대구 우동 1477 아이피파빌리온 3층 Tel: 051-606-1500
광주지사: 광주광역시 광산구 우산동 1589-1 광주무역회관 5층 Tel: 062-945-8666
대구지사: 대구광역시 북구 산격2동 1692번지 산업용재관 업무동 4층 Tel: 053-604-3960

Rockwell Automation Publication 1756-RM003T-KO-P - **2018 년 11 월**

Supersedes publication 1756-RM003S-KO-P - 2018 년 2 월