

# Rexroth IndraMotion MLC 12VRS 첫단 계

R911341385  
제01 판

가동 설명서



|         |                                                                                                                                                             |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 표제      | Rexroth IndraMotion<br>MLC 12VRS<br>첫 단계                                                                                                                    |
| 문서형태    | 가동 설명서                                                                                                                                                      |
| 문서-타입코드 | DOK-MLC***-F*STEP**V12-CO01-KO-P                                                                                                                            |
| 내부 파일문헌 | RS-bec1385da4450f050a6846a0005b156d-1-ko-KR-19                                                                                                              |
| 자료목적    | 이 설명서에서는 IndraMotion MLC 및 RobotControl 의 첫 단계에 대해 설명합니다. 여기에는 하드웨어 및 소프트웨어 사전 요구 사항뿐 아니라 프로젝트를 만드는 방법에 대해서도 나와 있습니다.<br>시스템의 온라인 도움말은 이 설명서를 바탕으로 작성되었습니다. |

|        |        |         |    |
|--------|--------|---------|----|
| 변경진행기록 | 간행물    | 공개 일시   | 비고 |
|        | 제 01 판 | 2014.03 | 초판 |

|      |                                                                                                                                                                                                                |
|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 저작권  | © Bosch Rexroth AG 2014<br>특허권 등록의 경우에도 모든 권리는 Bosch Rexroth AG 가 보유하며, 저작권 및 양도권과 같은 모든 처분 권한을 Bosch Rexroth AG 가 보유합니다.                                                                                      |
| 책임   | 명시된 데이터는 제품 설명 목적용으로만 작성되었으며, 계약서에 명확하게 규정되지 않은 한 보증된 특성으로 간주되지 않습니다. 이 문서의 내용 및 제품 사용과 관련된 모든 권리는 보호를 받습니다.                                                                                                   |
| 출판사  | Bosch Rexroth AG<br>Bgm.-Dr.-Nebel-Str. 2 • 97816 Lohr a. Main, Germany<br>전화 +49 (0)93 52 18 0 • 팩스 +49 (0)93 52 18 8400<br><a href="http://www.boschrexroth.com/">http://www.boschrexroth.com/</a><br>Gac2Lo |
| 주의사항 | 이 문서는 무염소 표백 용지에 인쇄되었습니다.                                                                                                                                                                                      |

# 내용목차

|          |                                      |           |
|----------|--------------------------------------|-----------|
| <b>1</b> | <b>설명서 정보.....</b>                   | <b>5</b>  |
| 1.1      | 설명서의 유효성.....                        | 5         |
| 1.2      | 필수 및 보충 설명서.....                     | 5         |
| <b>2</b> | <b>중요 사용 지침.....</b>                 | <b>11</b> |
| 2.1      | 적절한 사용.....                          | 11        |
| 2.1.1    | 소개.....                              | 11        |
| 2.1.2    | 사용 및 애플리케이션 영역.....                  | 11        |
| 2.2      | 부적절한 사용.....                         | 12        |
| <b>3</b> | <b>전기 드라이브 및 컨트롤 안전 지침 .....</b>     | <b>13</b> |
| 3.1      | 용어 정의.....                           | 13        |
| 3.2      | 일반 정보.....                           | 14        |
| 3.2.1    | 안전 지침의 사용과 다른 사람에게 안전 지침 전달.....     | 14        |
| 3.2.2    | 안전 사용을 위한 요구 사항.....                 | 14        |
| 3.2.3    | 부적절한 사용에 의한 위험.....                  | 15        |
| 3.3      | 특정 위험 관련 지침.....                     | 15        |
| 3.3.1    | 전기 부품 및 하우징에 접촉하지 않도록 보호.....        | 15        |
| 3.3.2    | 감전 방지를 위한 매우 낮은 보호 전압 .....          | 17        |
| 3.3.3    | 위험한 기계 작동으로부터 보호.....                | 17        |
| 3.3.4    | 작동 및 설치하는 동안 자기장 및 전자기장에 대비한 보호..... | 18        |
| 3.3.5    | 고온 부품과의 접촉으로부터 보호.....               | 19        |
| 3.3.6    | 취급 및 설치하는 동안 보호.....                 | 19        |
| 3.3.7    | 배터리 안전.....                          | 19        |
| 3.3.8    | 가압 시스템에 대한 보호.....                   | 20        |
| 3.4      | 신호 단어 및 안전 경고 기호에 대한 설명.....         | 20        |
| <b>4</b> | <b>첫 단계 - IndraLogic.....</b>        | <b>23</b> |
| 4.1      | 프로젝트 목표 "첫 단계 - IndraLogic".....     | 23        |
| 4.2      | 기술 요구 사항.....                        | 23        |
| 4.3      | 새 IndraWorks 프로젝트 만들기.....           | 24        |
| 4.3.1    | 컨트롤 재설정(CMLx5).....                  | 24        |
| 4.3.2    | IndraWorks 시작.....                   | 24        |
| 4.3.3    | 새 프로젝트 만들기.....                      | 24        |
| 4.4      | 프로젝트에 새 제품 추가.....                   | 25        |
| 4.5      | 제품에 기본 설정 할당.....                    | 26        |
| 4.5.1    | 제품설정.....                            | 26        |
| 4.5.2    | 통신 설정.....                           | 27        |
| 4.5.3    | 평선모듈설정.....                          | 28        |
| 4.6      | PLC 프로그램 만들기.....                    | 29        |
| 4.6.1    | 신호등 시스템의 컨트롤.....                    | 29        |
| 4.6.2    | 프로그램 및 평선블럭 만들기.....                 | 29        |
| 4.6.3    | 프로그램 및 평선블럭 편집 - 1 부.....            | 35        |

내용목차

|          | 페이지                                                |
|----------|----------------------------------------------------|
| 4.6.4    | 작업 구성..... 52                                      |
| 4.6.5    | 전역 변수 선언..... 53                                   |
| 4.6.6    | 시뮬레이션 모드에서 "SEQUENCE" 테스트..... 54                  |
| 4.6.7    | 프로그램 및 평선블럭 편집 - 2 부..... 57                       |
| 4.6.8    | 컨트롤에서 응용 프로그램 테스트..... 65                          |
| 4.7      | PLC 프로그램의 시각화 만들기..... 66                          |
| 4.7.1    | IndraLogic 의 시각화..... 66                           |
| 4.7.2    | 새 시각화 만들기..... 66                                  |
| 4.7.3    | 시각화 편집..... 67                                     |
| 4.7.4    | 시각화 테스트..... 77                                    |
| 4.8      | 프로젝트 복원..... 78                                    |
| <b>5</b> | <b>첫 단계 - 모션..... 81</b>                           |
| 5.1      | 프로젝트 목표 "첫 단계 - 모션"..... 81                        |
| 5.2      | 기술 요구 사항..... 81                                   |
| 5.3      | 모션 하위 프로젝트 만들기..... 82                             |
| 5.3.1    | 컨트롤 재설정(CMLx5)..... 82                             |
| 5.3.2    | IndraWorks 시작..... 82                              |
| 5.3.3    | 새 프로젝트 만들기..... 82                                 |
| 5.3.4    | 프로젝트에 새 제품 추가..... 83                              |
| 5.3.5    | 제품에 대한 기본 설정 지정..... 84                            |
| 5.3.6    | 프로젝트에 sercos III 드라이브(실제 축) 추가..... 87             |
| 5.3.7    | 프로젝트에 가상 축 추가..... 88                              |
| 5.3.8    | 모션 하위 프로젝트 다운로드..... 89                            |
| 5.4      | FBD(평선블럭다이어그램)에서 모션 스퀀스 프로그래밍..... 91              |
| 5.4.1    | FBD(평선블럭다이어그램)에서 모션 스퀀스 프로그래밍, 일반 정보..... 91       |
| 5.4.2    | 가상 축 VA1 프로그래밍, 프로그래밍 언어 FBD(평선블럭다이어그램)..... 93    |
| 5.4.3    | PLC 하위 프로젝트 가상 축 VA1 다운로드(FBD(평선블럭다이어그램))..... 100 |
| 5.4.4    | 축 VA1 의 모션 기록(트레이스), FBD(평선블럭다이어그램)..... 104       |
| 5.4.5    | 실제 축 RA1 프로그래밍(FBD(평선블럭다이어그램))..... 109            |
| 5.4.6    | 전체 PLC 프로젝트 다운로드(FBD(평선블럭다이어그램))..... 112          |
| 5.4.7    | 워크플로 "가상 축 → 전자식 기어 → 실제 축"..... 113               |
| 5.4.8    | 두 축의 모션 기록(트레이스), FBD(평선블럭다이어그램)..... 114          |
| 5.5      | ST(구조체언어)에서 모션 스퀀스 프로그래밍..... 115                  |
| 5.5.1    | ST(구조체언어)에서 모션 스퀀스 프로그래밍, 일반 정보..... 115           |
| 5.5.2    | 가상 축 VA1 프로그래밍, 프로그래밍 언어 ST(구조체언어)..... 117        |
| 5.5.3    | PLC 하위 프로젝트 가상 축 VA1 다운로드(ST(구조체언어))..... 123      |
| 5.5.4    | 축 VA1 의 모션 기록(트레이스), ST(구조체언어)..... 127            |
| 5.5.5    | 실제 축 RA1 프로그래밍(ST(구조체언어))..... 132                 |
| 5.5.6    | 전체 PLC 프로젝트 다운로드(ST(구조체언어))..... 136               |
| 5.5.7    | 워크플로 "가상 축 → 전자식 기어 → 실제 축"..... 136               |
| 5.5.8    | 두 축의 모션 기록(트레이스), ST(구조체언어)..... 138               |
| 5.6      | 프로젝트 복원..... 139                                   |

내용목차

페이지

|          |                                                |            |
|----------|------------------------------------------------|------------|
| <b>6</b> | <b>첫 단계 - FlexProfile.....</b>                 | <b>143</b> |
| 6.1      | 프로젝트 목표 "첫 단계 - FlexProfile".....              | 143        |
| 6.2      | 기술 요구 사항.....                                  | 144        |
| 6.3      | 모션 하위 프로젝트 만들기.....                            | 144        |
| 6.4      | FlexProfile 만들기.....                           | 145        |
| 6.4.1    | FlexProfile 만들기, 일반 정보.....                    | 145        |
| 6.4.2    | IndraWorks 대화 상자를 통해 FlexProfile 만들기.....      | 145        |
| 6.4.3    | PLC 를 통해 FlexProfile 만들기.....                  | 148        |
| 6.5      | PLC 하위 프로젝트 완료, FlexProfile 모션스텝 실행.....       | 151        |
| 6.6      | PLC 하위 프로젝트 다운로드 및 테스트.....                    | 155        |
| 6.7      | 워크플로 "가상 축 → FlexProfile 0 → 실제 축".....        | 155        |
| 6.8      | 두 축의 모션 기록(트레이스).....                          | 156        |
| 6.9      | 프로젝트 복원.....                                   | 157        |
| <b>7</b> | <b>첫 단계 - RobotControl.....</b>                | <b>159</b> |
| 7.1      | 프로젝트 목표 "첫 단계 - RobotControl".....             | 159        |
| 7.2      | 필수 하드웨어 및 소프트웨어.....                           | 159        |
| 7.3      | RoCo 프로젝트 만들기 절차 .....                         | 160        |
| 7.4      | "RoCoTestV12" 프로젝트에 대한 세부 정보.....              | 169        |
| 7.5      | RobotControl - 애플리케이션 예.....                   | 171        |
| 7.5.1    | 픽 앤 플레이스(Pick & Place) 애플리케이션의 RoCo 기본 사항..... | 171        |
| <b>8</b> | <b>서비스 및 지원.....</b>                           | <b>181</b> |
|          | <b>인덱스.....</b>                                | <b>183</b> |



# 1 설명서 정보

## 1.1 설명서의 유효성

- 대상 그룹** 이 설명서는 IndraMotion MLC 컨트롤을 사용하여 IndraWorks Engineering 시스템의 첫 단계를 시작하는 초보 사용자를 위해 작성되었습니다.  
이 설명서는 다음 장으로 구성되어 있습니다.
- [첫 단계 - IndraLogic, 23 페이지](#)  
IndraLogic 과 시각화를 사용한 프로그래밍에 대한 기본 사항  
이 장의 내용은 컨트롤을 사용하지 않는 시뮬레이션 모드(Soft-PLC)에도 적용할 수 있습니다.
  - [첫 단계 - 모션, 81 페이지](#)  
PLC 프로그램(PLCopen)의 드라이브 제어에 대한 기본 사항, 트레이스를 통해 축 모션 기록  
이 장의 내용은 가상 축을 사용하는 드라이브가 없을 때 실제 컨트롤을 사용하는 경우에도 적용할 수 있습니다.
  - [첫 단계 - FlexProfile, 143 페이지](#)  
모션 프로파일 만들기 및 적용  
이 장은 [첫 단계 - 모션](#)의 내용을 기반으로 합니다.
  - [첫 단계 - RobotControl, 159 페이지](#)  
RoCo 소스 코드를 실행할 때까지 RobotControl 프로젝트 만들기
- 일반 지침** 나머지 장에서는 다음에 대한 정보를 제공합니다.
- [필수 및 보충 설명서, 5 페이지](#)
  - [중요 사용 지침, 11 페이지](#) 및
  - [전기 드라이브 및 컨트롤 안전 지침, 13 페이지](#)

## 1.2 필수 및 보충 설명서

유형 지정 코드 및 부품 번호를 포함한 설명서 제목

| IndraWorks |                                                                                                                                                                                                                                      | MLC | XLC |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|
| /36/       | <b>Rexroth IndraWorks 12VRS Software Installation (Rexroth IndraWorks 12VRS 소프트웨어 설치)</b><br>DOK-IWORKS-SOFTINS*V12-COxx-EN-P, R911334396<br>이 설명서에서는 IndraWorks 설치 방법에 대해 설명합니다.                                                    | X   | X   |
| /5/        | <b>Rexroth IndraWorks 12VRS Engineering</b><br>DOK-IWORKS-ENGINEE*V12-APxx-EN-P, R911334388<br>이 설명서에서는 Rexroth Engineering 도구가 통합되는 IndraWorks 의 애플리케이션 분야에 대해 설명합니다. 여기에는 IndraWorks 를 사용하는 방법 및 오실로스코프 기능을 조작하는 방법에 대한 지침이 포함됩니다. | X   | X   |
| /20/       | <b>Rexroth IndraMotion MLC 12VRS Functional Description (Rexroth IndraMotion MLC 12VRS 기능 설명)</b><br>DOK-MLC***-FUNC****V12-APxx-EN-P, R911333848<br>이 설명서에서는 IndraMotion MLC 의 마법사, 상황에 맞는 메뉴, 대화 상자, 컨트롤 가동, 제품 구성 및 기능에 대해 설명합니다. | X   |     |

설명서 정보

|      |                                                                                                                                                                                                                                                                                                                                         |   |   |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|---|
| /20/ | <b>Rexroth IndraLogic XLC 12VRS Functional Description(Rexroth IndraLogic XLC 12VRS 기능 설명)</b><br>DOK-XLC***-FUNC***V12-APxx-EN-P, R911333878<br>이 설명서에서는 IndraLogic XLC 의 마법사, 상황에 맞는 메뉴, 대화 상자, 컨트롤 가동, 제품 구성 및 기능에 대해 설명합니다.                                                                                                         |   | X |
| /7/  | <b>Rexroth IndraWorks 12VRS CamBuilder</b><br>DOK-IWORKS-CAMBUIL*V12-APxx-EN-P, R911333842<br>이 설명서에서는 캠 편집 도구인 CamBuilder 의 기본 원리 및 작동 방법에 대해 설명합니다.                                                                                                                                                                                   | X | X |
| /37/ | <b>Rexroth IndraLogic XLC IndraMotion MLC 12VRS Automation Interface (Rexroth IndraLogic XLC IndraMotion MLC 12VRS 자동화 인터페이스)</b><br>DOK-XLCMLC-AUT*INT*V12-APxx-EN-P, R911334178<br>이 설명서에서는 자동화 인터페이스의 인터페이스를 통해 스크립트를 기반으로 IndraWorks 프로젝트 데이터에 액세스하는 방법에 대해 설명합니다.                                                                  | X | X |
| /38/ | <b>Rexroth IndraWorks 12VRS FDT Container (Rexroth IndraWorks 12VRS FDT 컨테이너)</b><br>DOK-IWORKS-FDT*CON*V12-APxx-EN-P, R911334398<br>이 설명서에서는 IndraWorks FDT 컨테이너의 기능에 대해 설명합니다. 프로젝트에서 기능을 활성화하고 DTM 을 사용하는 방법이 제공됩니다.                                                                                                                 | X | X |
| /29/ | <b>Rexroth IndraLogic XLC IndraMotion MLC 12VRS Project Conversion (Rexroth IndraLogic XLC IndraMotion MLC 12VRS 프로젝트 변환)</b><br>DOK-XLCMLC-PROCONV*V12-APxx-EN-P, R911334187<br>이 설명서에서는 IndraWorks 버전 12(IndraLogic 2G 포함)에서 IndraLogic 04VRS 및 IndraMotion MLC04VRS 의 프로젝트 변환에 대해 설명합니다. 특히 Motion 및 PLC 의 필드 변경 사항에 대해 집중적으로 설명합니다. | X | X |
| /28/ | <b>Rexroth IndraMotion MLC 12VRS Commissioning (Rexroth IndraMotion MLC 12VRS 가동)</b><br>DOK-MLC***-STARTUP*V12-COxx-EN-P, R911333858<br>이 설명서에서는 IndraMotion MLC 시스템을 가동하고 이 시스템에 대한 서비스를 수행하는 데 필요한 단계에 대해 설명합니다. 여기에는 자주 수행하는 작업에 대한 검사 목록 및 단계에 대한 자세한 설명이 나와 있습니다.                                                                 | X |   |

| Motion |                                                                                                                                                                                                                                                                                                                                          | MLC | XLC |
|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|
| /23/   | <b>Rexroth IndraLogic XLC IndraMotion MLC 12VRS PLCopen Libraries (Rexroth IndraLogic XLC IndraMotion MLC 12VRS PLCopen 라이브러리)</b><br>DOK-XLCMLC-FUNLIB**V12-LIxx-EN-P, R911334182<br>이 설명서에서는 IndraLogic XLC/IndraMotion MLC 에 대한 RIL_CommonTypes, ML_Base 및 ML_PLCopen 라이브러리의 평선블럭, 기능 및 데이터 유형에 대해 설명합니다. 또한 평선블럭의 오류 반응에 대해서도 설명합니다. | X   | X   |
| /27/   | <b>Rexroth IndraLogic XLC IndraMotion MLC 12VRS Generic Application Template(Rexroth IndraLogic XLC IndraMotion MLC 12VRS 일반 응용 프로그램 템플릿)</b><br>DOK-XLCMLC-TF*GAT**V12-APxx-EN-P, R911334191<br>이 설명서에서는 IndraLogic PLC 프로그래머에게 구조화된 템플릿을 제공합니다. 이 템플릿을 사용하면 PLC 프로그래밍 코드를 추가하고 편집할 수 있습니다. 여기에는 템플릿, 템플릿 마법사 및 예제 응용 프로그램이 포함되어 있습니다.  | X   | X   |
| /31/   | <b>Rexroth IndraMotion MLC 12VRS RCL Programming Instruction(Rexroth IndraMotion MLC 12VRS RCL 프로그래밍 지침)</b><br>DOK-MLC***-RCL*PRO*V12-APxx-EN-P, R911333852<br>이 설명서에서는 RobotControl 에 대한 정보를 제공합니다. 여기에서는 프로그래밍 언어 RCL(RobotControl Language, 로봇 제어 언어)을 집중적으로 다루며, 프로그램 구조, 변수, 함수, 모션 명령문 및 필수 시스템 매개변수에 대해 설명합니다.                   | X   |     |

설명서 정보

|      |                                                                                                                                                                                                                                                                                                                         |   |   |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|---|
| /21/ | <b>Rexroth IndraLogic XLC IndraMotion MLC 12VRS Parameters (Rexroth IndraLogic XLC IndraMotion MLC 12VRS 매개변수)</b><br>DOK-XLCMLC-PARAM***V12-RExx-EN-P, R911334176<br>이 설명서에서는 XLC/MLC 시스템의 매개변수뿐 아니라 매개변수 입력 및 프로그래밍 간의 상호 작용에 대해 설명합니다. 여기에는 축 매개변수, 컨트롤 매개변수, 키네마틱 매개변수, 터치 프로브 매개변수 및 프로그래밍 가능한 제한 스위치 매개변수가 포함됩니다. | X | X |
| /10/ | <b>Rexroth IndraDrive; Firmware for Drive Controllers MPH, MPB, MPD, MPC-07(Rexroth IndraDrive; 드라이브 컨트롤러 MPH, MPB, MPD, MPC-07 용 펌웨어)</b><br>DOK-INDRV*-MP*-07VRS**-FKxx-EN-P, R911328670                                                                                                                              |   |   |
| /11/ | <b>Rexroth IndraDrive MPx-16 Functions (Rexroth IndraDrive MPx-16 기능)</b><br>DOK-INDRV*-MP*-16VRS**-APxx-EN-P, R911326767                                                                                                                                                                                               |   |   |

| 필드 버스 |                                                                                                                                                                                                                                                                                                                                                                             | MLC | XLC |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|
| /39/  | <b>Rexroth IndraMotion MLC 11VRS PLCopen Field Bus (Rexroth IndraMotion MLC 11VRS PLCopen 필드 버스)</b><br>DOK-IM*ML*-PLCFBUS*V11-APxx-EN-P, R911333896<br>이 설명서에서는 IndraWorks 프로젝트에서 필드 버스 드라이브 만들기 및 RIL_CommonTypes.library(필드 버스 드라이브 제외), RMB_PLcopenFieldBus.library, RIL_Utility.library(필드 버스 드라이브 제외) 라이브러리의 평선블록, 기능 및 데이터 유형에 대해 설명합니다. 또한 평선블록의 오류 반응에 대해서도 설명합니다. | X   | X   |
| /4/   | <b>Rexroth IndraLogic XLC IndraMotion MLC 12VRS Field Buses (Rexroth IndraLogic XLC IndraMotion MLC 12VRS 필드 버스)</b><br>DOK-XLCMLC-FB*****V12-APxx-EN-P, R911334394<br>이 설명서에서는 지원되는 필드 버스 및 해당 필드 버스의 진단 평선블록에 대해 설명합니다.                                                                                                                                                   | X   | X   |

| HMI  |                                                                                                                                                                                                                                                              | MLC | XLC |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|
| /8/  | <b>Rexroth IndraWorks 12VRS HMI</b><br>DOK-IWORKS-HMI*****V12-APxx-EN-P, R911334392<br>이 설명서에서는 사용자 인터페이스 IndraWorks HMI Engineering 및 IndraWorks HMI Operation 의 기능, 구성 및 작동에 대해 설명합니다.                                                                     | X   | X   |
| /6/  | <b>Rexroth IndraWorks 12VRS WinStudio</b><br>DOK-IWORKS-WINSTUD*V12-APxx-EN-P, R911333844<br>이 설명서에서는 소프트웨어 설치, WinStudio 사용, 응용 프로그램 만들기 및 조작에 대해 설명합니다.                                                                                                    | X   | X   |
| /50/ | <b>Rexroth IndraLogic XLC IndraMotion MLC 12VRS HMI Connection (Rexroth IndraLogic XLC IndraMotion MLC 12VRS HMI 연결)</b><br>DOK-XLCMLC-HMI*****V12-APxx-EN-P, R911334184<br>이 설명서에서는 IndraLogic XLC 및 IndraMotion MLC 에서 지원하는 시각화 시스템과 해당 시스템의 연결에 대해 설명합니다. | X   | X   |

## 설명서 정보

| PLC  |                                                                                                                                                                                                                                                                                           | MLC | XLC |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|
| /3/  | <b>Rexroth IndraWorks 12VRS IndraLogic 2G Programming Instruction (Rexroth IndraWorks 12VRS IndraLogic 2G 프로그래밍 지침)</b><br>DOK-IWORKS-IL2GPRO*V12-APxx-EN-P, R911334390<br>이 설명서에서는 PLC 프로그래밍 도구인 IndraLogic 2G 와 그 사용법에 대해 설명합니다. 여기에는 기본 사용, 첫 번째 단계, 시각화, 메뉴 항목 및 편집기 등에 대한 내용이 포함됩니다. | X   | X   |
| /33/ | <b>Rexroth IndraWorks 12VRS, Basic Libraries, IndraLogic 2G (Rexroth IndraWorks 12VRS, 기본 라이브러리, IndraLogic 2G)</b><br>DOK-IL*2G*-BASLIB**V12-LIxx-EN-P, R911333835<br>이 설명서에서는 시스템 전반의 PLC 라이브러리에 대해 설명합니다.                                                                              | X   | X   |

| 테크놀로지 |                                                                                                                                                                                                                                                                                                                                                                                             |   |  |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|--|
| /30/  | <b>Rexroth IndraMotion MLC 12VRS Technology Libraries (Rexroth IndraMotion MLC 12VRS 기술 라이브러리)</b><br>DOK-MLC***-TF*LIB**V12-LIxx-EN-P, R911333868<br>이 설명서에서는 "ML_TechInterface.library", "ML_TechMotion.library", "RMB_TechCam.library" 및 "ML_TechBase.library" 라이브러리의 평선블럭, 기능 및 데이터 유형에 대해 설명합니다. 또한 여기에는 와인더 기능, 레지스터 컨트롤러 기능 및 크로스 커터 기능에 대한 라이브러리도 포함됩니다.                            | X |  |
| /60/  | <b>Rexroth IndraMotion MLC 12VRS RegisterControl (Library) (Rexroth IndraMotion MLC 12VRS 레지스터 컨트롤러(라이브러리))</b><br>DOK-MLC***-REGI*CO*V12-LIxx-EN-P, R911333856<br>이 설명서에서는 각 평선블럭의 입력 및 출력에 대해 설명하고 사용법에 대한 주의사항을 제공합니다.                                                                                                                                                                   | X |  |
| /62/  | <b>Rexroth IndraMotion MLC 12VRS RegisterControl (Application Manual) (Rexroth IndraMotion MLC 12VRS 레지스터 컨트롤러(응용 프로그램 설명서))</b><br>DOK-MLC***-REGI*CO*V12-APxx-EN-P, R911333854<br>이 설명서에서는 운전 그라비아 인쇄 머신에 대한 통합 레지스터 컨트롤의 애플리케이션에 대해 설명합니다. 마크 스트림 센서의 구성 요소, HMI 응용 프로그램 및 오류 복구 옵션에 대해서도 설명합니다. 이 지침은 레지스터 컨트롤 조작, 오류에 반응 및 진단 쿼리 방법에 대한 정보를 제공합니다. 이 설명서는 머신 가동자 및 머신 조작자를 위해 작성되었습니다. | X |  |
| /49/  | <b>Rexroth IndraMotion MLC 12VRS Application of Winder Functions (Rexroth IndraMotion MLC 12VRS 와인더 기능의 애플리케이션)</b><br>DOK-MLC***-TF*WIND*V12-APxx-EN-P, R911333870<br>이 애플리케이션 관련 시스템 설명서에서는 와인더 기술의 기능을 사용하는 방법에 대해 설명합니다.                                                                                                                                                                | X |  |

| 하드웨어 |                                                                                                                                                                                                       | MLC | XLC |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|
| /1/  | <b>Rexroth IndraControl L45/L65</b><br>DOK-CONTRL-IC*L45*L65*-PRxx-EN-P, R911324661                                                                                                                   | X   | X   |
| /2/  | <b>Rexroth IndraControl L25</b><br>DOK-CONTRL-IC*L25*****-PRxx-EN-P, R911328474                                                                                                                       | X   | X   |
| /24/ | <b>Rexroth IndraControl Lxx 12VRS Function Modules (Rexroth IndraControl Lxx 12VRS 평선모듈)</b><br>DOK-CONTRL-FM*LXX**V12-APxx-EN-P, R911333830<br>이 설명서에서는 엔지니어링 및 진단을 포함하여 Lxx 컨트롤의 모든 평선모듈에 대해 설명합니다. | X   | X   |
| /12/ | <b>Rexroth IndraDrive Drive Controllers MPx-02 to MPx-07 (Rexroth IndraDrive 드라이브 컨트롤러 MPx-02 ~ MPx-07)</b><br>DOK-INDRV*-GEN-**VRS**-PAxx-EN-P, R911297317                                           |     |     |

설명서 정보

|      |                                                                                                                                                                                |  |  |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|
| /13/ | Rexroth IndraDrive MPx-02 to MPx-07 and HMV (Rexroth IndraDrive MPx-02 ~ MPx-07 및 HMV)<br>DOK-INDRV*-GEN-**VRS**-WAXx-EN-P, R911297319                                         |  |  |
| /35/ | Rexroth IndraDrive Drive Controller Control Sections CSB01, CSH01, CDB01(Rexroth IndraDrive 드라이브 컨트롤러 컨트롤 섹션 CSB01, CSH01, CDB01)<br>DOK-INDRV*-CSH*****-PR08-EN-P, R911295012 |  |  |

| 진단 및 서비스 |                                                                                                                                                                                                                                              | MLC | XLC |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|
| /26/     | Rexroth IndraMotion MLC/XLC 11VRS Service Tool (Rexroth IndraMotion MLC/XLC 11VRS 서비스 도구)<br>DOK-IM*ML*-IMST***V11-RExx-EN-P, R911331940                                                                                                     | X   | X   |
| /22/     | Rexroth IndraLogic XLC IndraMotion MLC 12VRS Diagnostics (Rexroth IndraLogic XLC IndraMotion MLC 12VRS 진단)<br>DOK-XLCMLC-DIAG***V12-RExx-EN-P, R911334180<br>이 설명서에는 컨트롤 시스템 IndraLogic XLC 및 IndraMotion MLC 에서 구현된 모든 컨트롤 매개변수가 포함되어 있습니다. | X   | X   |

| 시스템 개요 |                                                                                                                                                                                                                                                                            | MLC | XLC |
|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|
| /48/   | Rexroth IndraMotion for Printing 12VRS System Overview (Rexroth IndraMotion for Printing 12VRS 시스템 개요)<br>DOK-IM*PR*-SYSTEM**V11-PRxx-EN-P, R911333840<br>이 설명서에는 IndraMotion for Packaging 제품에 대해 설명합니다. 여기에서는 컨트롤 시스템, 드라이브 시스템 및 I/O 시스템뿐 아니라 가동 및 프로그래밍에 대해서도 소개합니다.   | X   |     |
| /48/   | Rexroth IndraMotion for Packaging 12VRS System Overview (Rexroth IndraMotion for Packaging 12VRS 시스템 개요)<br>DOK-IM*PA*-SYSTEM**V12-PRxx-EN-P, R911333838<br>이 설명서에는 IndraMotion for Packaging 제품에 대해 설명합니다. 여기에서는 컨트롤 시스템, 드라이브 시스템 및 I/O 시스템뿐 아니라 가동 및 프로그래밍에 대해서도 소개합니다. | X   |     |
| /9/    | Rexroth IndraMotion MLC 12VRS System Overview (Rexroth IndraMotion MLC 12VRS 시스템 개요)<br>DOK-MLC***-SYSTEM**V12-PRxx-EN-P, R911333860<br>이 설명서에서는 지정된 버전의 자동화 시스템 IndraMotion MLC 에 사용 가능한 하드웨어/소프트웨어 구성 요소에 대해 개략적으로 설명합니다. 또한 이 설명서는 시스템을 조립하는 데에도 유용합니다.                 | X   |     |
| /9/    | Rexroth IndraLogic XLC 12VRS System Overview (Rexroth IndraLogic XLC 12VRS 시스템 개요)<br>DOK-XLC***-SYSTEM**V12-PRxx-EN-P, R911333880<br>이 설명서에서는 지정된 버전의 자동화 시스템 IndraLogic XLC 에 사용 가능한 하드웨어/소프트웨어 구성 요소에 대해 개략적으로 설명합니다. 또한 이 설명서는 시스템을 조립하는 데에도 유용합니다.                    |     | X   |

| 첫 단계 |                                                                                                                                                                                                                                                         | MLC | XLC |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|
| /25/ | Rexroth IndraMotion MLC 12VRS First Steps (Rexroth IndraMotion MLC 12VRS 첫 단계)<br>DOK-MLC***-F*STEP**V12-COxx-EN-P, R911333846<br>이 설명서에서는 IndraMotion MLC 및 RobotControl 의 첫 단계에 대해 설명합니다. 여기에는 하드웨어 및 소프트웨어 사전 요구 사항뿐 아니라 프로젝트를 만드는 방법에 대해서도 나와 있습니다. | X   |     |
| /25/ | Rexroth IndraLogic XLC 12VRS First Steps (Rexroth IndraLogic XLC 12VRS 첫 단계)<br>DOK-XLC***-F*STEP**V12-COxx-EN-P, R911333876<br>이 설명서에서는 IndraLogic XLC 의 첫 단계에 대해 설명합니다. 여기에는 하드웨어 및 소프트웨어 사전 요구 사항뿐 아니라 프로젝트를 만드는 방법에 대해서도 나와 있습니다.                   |     | X   |



## 2 중요 사용 지침

### 2.1 적절한 사용

#### 2.1.1 소개

Rexroth 제품은 최신 기술을 사용하여 개발 및 제조되었으며 작동 안전 및 신뢰성을 보장하기 위해 제공 전에 테스트됩니다.

Rexroth 제품은 적절한 방식으로만 사용할 수 있습니다. 적절한 방식으로 사용하지 않을 경우 제품이 손상되고 부상을 입을 수 있습니다.



제조업체로서 Bosch Rexroth는 부적절한 사용으로 인한 어떠한 손해에 대해서도 책임을 지지 않습니다. 이러한 경우 부적절한 사용으로 인한 손해 배상에 대한 보증과 권한이 배제됩니다. 위험에 대한 모든 책임은 전적으로 사용자에게 있습니다.

Rexroth 제품을 사용하기 전에 제품의 적절한 사용을 위한 모든 필요 조건을 충족하는지 확인합니다.

- 어떠한 방식이나 형태로든 제품을 사용하는 사람은 먼저 관련 안전 지침을 읽고 숙지해야 하며 적절한 사용에 친숙해져야 합니다.
- 제품이 하드웨어 형태를 가진 경우 원래 상태를 유지해야 합니다(즉, 구조적 변경이 허용되지 않음). 소프트웨어 제품을 디컴파일하거나 소스 코드를 변경하는 것은 허용되지 않습니다.
- 손상되었거나 결함 있는 제품을 탑재하거나 작동 중에 사용하지 마십시오.
- 관련 설명서에 설명된 대로 제품을 설치했는지 확인하십시오.

#### 2.1.2 사용 및 애플리케이션 영역

Rexroth IndraMotion MLC

IndraMotion MLC 및 해당 평션모듈은 모션/로직 응용 프로그램용입니다.



IndraMotion MLC 및 해당 평션모듈에는 이 설명서에 나열된 악세사리 및 장착 부품만 사용할 수 있습니다. 명시적으로 언급되지 않은 구성 요소는 부착하거나 연결해서는 안 됩니다. 케이블과 라인의 경우도 마찬가지입니다.

명시적으로 언급된 하드웨어 구성 요소 구성 및 조합과 각 설명서 및 기능 설명에 지정된 소프트웨어 및 펌웨어로만 작업을 수행해야 합니다.

IndraMotion MLC 및 해당 평션모듈은 단일 및 다중 축 드라이브 및 제어 작업에 사용하기 위해 개발되었습니다.

머신 모션 및 시각화 터미널의 애플리케이션 전용을 위해 각각 장비 및 인터페이스가 다른 제품 모델을 사용할 수 있습니다.

IndraMotion MLC 및 해당 평션모듈을 적용할 수 있는 일반적인 사용 범위는 다음과 같습니다.

- [취급 및 탑재 시스템]
- [포장 및 식품 가공 머신]
- [인쇄 및 제지 처리 머신]

Rexroth IndraMotion MLC 및 평션모듈은 관련 설명서에 지정된 탑재 및 설치 조건, 위치 및 대기조건(온도, 보호 유형, 습도, EMC 등)에서만 조작해야 합니다.

## 중요 사용 지침

## 2.2 부적절한 사용

Rexroth IndraMotion MLC 및 해당 펌션모듈을 설명서 및 기술 데이터에 지정되어 있거나 설명된 것과 다른 애플리케이션 분야에 사용할 경우 "부적절한 사용"으로 간주됩니다.

다음과 같은 경우에는 Rexroth IndraMotion MLC 및 해당 펌션모듈을 사용할 수 없습니다.

- 지정된 대기조건과 일치하지 않는 작동 조건에 노출된 경우. 물 속이나 온도 변화가 심한 곳이나 과도한 온도에 도달하는 곳에서는 작동이 금지됩니다.
- 또한 Bosch Rexroth 가 명시적으로 승인하지 않은 애플리케이션 분야에서는 IndraMotion MLC 및 해당 펌션모듈을 사용해서는 안 됩니다. 따라서 기본안전지침에 나와 있는 내용을 자세히 읽어 보십시오.

## 3 전기 드라이브 및 컨트롤 안전 지침

### 3.1 용어 정의

|                    |                                                                                                                                                                                                                                                                                                                                                                |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>예폴리케이션 자료</b>   | 적용 설명서는 제품 구성, 통합, 설치, 탑재, 가동, 작동, 정비, 수리 및 가동 해제를 위한 사용 및 안전 관련 기능을 제품 사용자에게 알리는 데 사용되는 전체 설명서로 구성됩니다. 또한 이 종류의 설명서에는 사용자 안내서, 작동 설명서, 가동 설명서, 지침 설명서, 프로젝트 계획 설명서 등의 용어가 사용됩니다.                                                                                                                                                                              |
| <b>구성 요소</b>       | 구성 요소는 장비, 장치 또는 시스템의 일부인 지정된 기능을 가진 요소의 조합입니다. 예를 들어, 전기 드라이브 및 컨트롤 시스템의 구성 요소는 공급 장치, 드라이브 컨트롤러, 전원 쇼크, 전원 필터, 모터, 케이블 등입니다.                                                                                                                                                                                                                                 |
| <b>컨트롤 시스템</b>     | 컨트롤 시스템은 단일 기능 장치로 시장에 출시되는 상호 연결된 여러 컨트롤 구성 요소로 구성됩니다.                                                                                                                                                                                                                                                                                                        |
| <b>제품</b>          | 장치는 개별 상품으로 시장에 출시되는 사용자를 위한 정의된 기능을 가진 완성된 제품입니다.                                                                                                                                                                                                                                                                                                             |
| <b>전기 장비</b>       | 전기 장비는 전기 모터, 트랜스, 전환 장치, 케이블, 라인, 전력 소비 장치, 회로 보드 어셈블리, 플러그인 장치, 컨트롤 캐비닛 등과 같이 전기 에너지를 생성, 변환, 전송, 분배 또는 적용하는 데 사용되는 모든 장치를 포함합니다.                                                                                                                                                                                                                            |
| <b>전기 드라이브 시스템</b> | 전기 드라이브 시스템은 전원 공급 장치에서 모터 샤프트에 이르기까지 모든 구성 요소를 포함합니다. 예를 들어, 전기 모터, 모터 엔코더, 공급 장치, 드라이브 컨트롤러뿐만 아니라 전원 필터, 전원 스로틀 및 해당 라인과 케이블 등의 보조 및 추가 구성 요소가 포함됩니다.                                                                                                                                                                                                        |
| <b>설치</b>          | 설치는 정의된 목적을 위해 정의된 사이트에 있는 상호 연결된 여러 장치 또는 시스템으로 구성됩니다. 그러나 이러한 장치 또는 시스템을 단일 기능 장치로 시장에 출시하기 위한 것은 아닙니다.                                                                                                                                                                                                                                                      |
| <b>머신</b>          | 장비는 상호 연결된 부품 또는 장치의 전체이며 이러한 부품 또는 장치 중 하나 이상을 이동할 수 있습니다. 따라서 장비는 적절한 장비 드라이브 요소뿐만 아니라 특정 적용을 위해 조립된 컨트롤 및 전원 회로로 구성됩니다. 예를 들어, 장비는 재료의 처리, 취급, 이동 또는 포장에 사용될 수 있습니다. 또한 "장비"이라는 용어는 하나의 통합된 단위로 작동하도록 정렬 및 제어되는 장비 조합을 의미합니다.                                                                                                                               |
| <b>제조업체</b>        | 제조업체는 개인 및 법적 단체의 이름으로 시장에 출시되는 제품의 디자인 및 제조를 책임지는 개인 또는 법적 단체입니다. 제조업체는 완성된 제품, 완성된 부품 또는 완성된 요소를 사용하거나 하도급 계약을 체결할 수 있습니다. 그러나 제조업체는 항상 제품을 전반적으로 제어하고 제품에 필요한 책임을 져야 합니다.                                                                                                                                                                                   |
| <b>제품</b>          | 제품의 예로는 장치, 구성 요소, 부품, 시스템, 소프트웨어, 펌웨어 등이 있습니다.                                                                                                                                                                                                                                                                                                                |
| <b>프로젝트 계획 설명서</b> | 프로젝트 계획 설명서는 시스템, 장비 또는 설치의 크기 조정 및 계획을 지원 하는 데 사용되는 적용 설명서의 일부입니다.                                                                                                                                                                                                                                                                                            |
| <b>책임자</b>         | 이 적용 설명서의 용어에서 책임자는 전기 드라이브 및 컨트롤 시스템의 구성 요소 설치, 탑재, 가동 및 작동뿐만 아니라 이에 따른 위험에 익숙하고 작업에 필요한 자격을 보유한 사람입니다. 이러한 자격을 갖추려면 특히 다음이 필요합니다. <ol style="list-style-type: none"> <li>1) 전기 회로와 장치 켜기 및 끄기, 접지, 마킹 등을 안전하게 수행할 수 있도록 교육 또는 지시를 받거나 권한을 부여 받아야 합니다.</li> <li>2) 적절한 안전 장비를 정비 및 사용할 수 있도록 교육 또는 지시를 받아야 합니다.</li> <li>3) 응급 처치 교육 과정에 참석해야 합니다.</li> </ol> |
| <b>사용자</b>         | 사용자는 시장에 출시된 제품을 설치, 가동 또는 사용하는 사람입니다.                                                                                                                                                                                                                                                                                                                         |

## 전기 드라이브 및 컨트롤 안전 지침

## 3.2 일반 정보

### 3.2.1 안전 지침의 사용과 다른 사람에게 안전 지침 전달

제품과 함께 제공된 모든 설명서를 먼저 읽은 후에 전기 드라이브 및 컨트롤 시스템의 구성 요소를 설치 및 작동하십시오. 이러한 구성 요소를 작동하기 전에 안전 지침과 모든 사용 설명서를 잘 읽고 이해하십시오. 구성 요소의 사용 설명서가 없으면 해당 Bosch Rexroth 판매 담당자에게 연락하여 구성 요소의 안전한 작동을 책임지고 있는 직원에게 설명서를 즉시 송부하도록 요청하시기 바랍니다.

구성 요소를 재판매, 임대 및/또는 기타 모든 형태로 다른 사람에게 넘긴 경우에는 사용자 국가의 공식 언어로 된 안전 지침을 구성 요소와 함께 전달해야 합니다.

구성 요소의 부적절한 사용, 이 설명서의 안전 지침 미준수 또는 안전 장치 비활성화를 포함하여 제품을 마음대로 변경하는 행위를 할 경우 제품 손상, 부상, 감전이나 심지어 사망까지 발생할 수 있습니다.

### 3.2.2 안전 사용을 위한 요구 사항

부상 및/또는 제품 손상의 위험을 방지하기 위해 전기 드라이브 및 컨트롤 시스템의 구성 요소를 처음 가동하기 전에 다음 지침을 읽으십시오. 이러한 안전 지침을 따라야 합니다.

- Bosch Rexroth 는 안전 지침을 지키지 않아서 발생한 손해에 대해 책임지지 않습니다.
- 가동하기 전에 해당 언어로 된 작동, 정비 및 안전 지침을 꼭 읽어보십시오. 해당 언어로 된 적용 설명서를 완전히 이해할 수 없다고 생각되면 공급자에게 분명히 확인하시기 바랍니다.
- 구성 요소를 작동하고 정비할 때 주의를 기울여야 할 뿐 아니라 적절하고 올바르게 운반, 보관, 탑재 및 설치하는 것 또한 이 구성 요소를 안전하게 최적의 상태로 작동하기 위한 선결 조건입니다.
- 책임자만 전기 드라이브 및 컨트롤 시스템의 구성 요소나 그 근처에 있는 구성 요소로 작업할 수 있습니다.
- Bosch Rexroth 에서 승인한 부속품 및 예비 부품만을 사용할 수 있습니다.
- 전기 드라이브 및 제어 시스템의 구성 요소가 작동하는 국가의 안전 규정 및 요구 사항을 따릅니다.
- 적절하게 정의된 방식으로 전기 드라이브 및 컨트롤 시스템의 구성 요소만 사용합니다. "적절한 사용" 장을 참조하십시오.
- 사용 가능한 적용 설명서에 명시된 주변 및 작동 조건을 준수해야 합니다.
- 기능 안전을 위한 적용은 적용 설명서 "통합 안전 기술"에 분명하고 명시적으로 지정된 경우에만 허용됩니다. 그렇지 않으면 사용이 배제됩니다. 기능 안전은 전기, 전자 또는 프로그래밍 가능한 컨트롤 시스템에 따라 개인 안전을 위한 위험 감소 조치가 달라지는 안전 개념입니다.
- 제공된 구성 요소 사용과 관련하여 적용 설명서에 명시된 정보에는 적용 및 제안 사례만 포함됩니다.

따라서 장비 및 설치 제조업체는 다음 사항을 수행해야 합니다.

- 제공된 구성 요소가 사용자의 개별 적용에 적합한지 확인하고 구성 요소 사용에 관해 이 적용 설명서에 나온 정보를 확인합니다.
- 개별 적용에서 해당 안전 규정 및 표준을 준수하는지 확인하고 필요한 조치, 수정 및 보완 사항을 수행합니다.

## 전기 드라이브 및 컨트롤 안전 지침

- 구성 요소가 설치된 장비 또는 설치가 해당국의 규정, 적용의 안전 사양 및 표준을 확실히 준수하는 경우에만 제공된 구성 요소를 가동할 수 있습니다.
- 또한 적용에 대한 국가 EMC 규정을 충족해야만 작동할 수 있습니다.
- EMC 요구 사항에 따른 설치 지침은 해당 적용 설명서의 EMC 섹션에 나와 있습니다.  
장비 또는 설치 제조업체는 국가 규정에 미리 정해진 제한값을 준수할 의무가 있습니다.
- 구성 요소의 기술 데이터, 연결 및 설치 조건은 해당 적용 설명서에 명시되어 있으며 이를 항상 준수해야 합니다.

### 사용자가 반드시 고려해야 하는 국가 규정

- 유럽 국가: 유럽 EN 표준 준수
- 미국:
  - NEC(National Electrical Code: 미국전기공사규정)
  - NEMA(National Electrical Manufacturers Association: 미국전기공업협회)와 지역 엔지니어링 규정
  - NFPA(National Fire Protection Association: 미국화재예방협회) 규정
- 캐나다: CSA(Canadian Standards Association: 캐나다 표준 협회)
- 그외 국가:
  - ISO(International Organization for Standardization: 국제 표준화 기구)
  - IEC(International Electrotechnical Commission: 국제전기회의)

### 3.2.3 부적절한 사용에 의한 위험

- 고전압 및 고압의 작동 전류! 감전에 의한 사망이나 심각한 부상 위험!
- 잘못된 연결에 의한 고전압! 감전에 의한 사망이나 부상 위험!
- 위험한 기계 작동! 의도하지 않은 모터 작동에 의한 생명 위험, 심각한 부상 또는 제품 손상!
- 전기 드라이브 시스템 근처에서 심박 작동기, 금속 삽입물이나 보청기를 착용한 사람의 경우 건강상 위험!
- 고온 하우징 표면에 의한 화상 위험!
- 부적절한 취급으로 인한 부상 위험! 타박상, 전단, 절개 및 타격으로 인한 부상!
- 배터리의 부적절한 취급으로 인한 부상 위험!
- 가압 라인의 부적절한 취급으로 인해 생기는 부상 위험!

## 3.3 특정 위험 관련 지침

### 3.3.1 전기 부품 및 하우징에 접촉하지 않도록 보호



이 섹션에서는 **50V 이상의 전압**이 있는 전기 드라이브 및 컨트롤 시스템의 구성 요소에 대한 내용을 다룹니다.

50V 이상 전압을 전도하는 부품에 닿으면 신체 위험 및 감전 사고가 일어날 수 있습니다. 전기 드라이브 및 컨트롤 시스템의 구성 요소를 작동할 때는 이러한 구성 요소의 일부 부품이 불가피하게 위험한 전압을 유도하게 됩니다.

## 전기 드라이브 및 컨트롤 안전 지침

**고전압! 감전 또는 심각한 부상으로 인한 생명 위험, 부상 위험!**

- 책임자만 전기 드라이브 및 컨트롤 시스템의 구성 요소를 작동, 정비 및/또는 수리할 수 있습니다.
- 전원 설치 작업 시에는 일반 설치 및 안전 규정을 준수하십시오.
- 장치를 켜기 전에 장비 연결도에 따라 장비 접지 단자가 모든 전기 구성 요소와 영구적으로 연결되어 있어야 합니다.
- 장비 접지 단자가 이 용도로 제공된 구성 요소의 지점과 영구적으로 연결되어 있지 않으면 간단한 측정이나 테스트를 하는 짧은 순간이라도 전기 장비를 작동하면 안 됩니다.
- 50V 이상의 전압 전위를 띠는 전기 부품에 접근하기 전에 전원이나 전원 공급 장치와의 전기 구성 요소 연결을 해제해야 합니다. 전기 구성 요소가 다시 연결되지 않게 하십시오.
- 전기 구성 요소에 대해서는 다음 사항을 준수하십시오.  
전기 구성 요소에 접근하기 전에 활성 콘덴서가 모두 방전되도록 전원을 끈 다음 **30 분** 기다립니다. 장비에 닿아도 안전한지 확인하기 위해 다시 작동을 시작하기 전에 활성 부품의 전기 전압을 측정합니다.
- 전원을 켜기 전에 이 목적을 위해 제공된 덮개와 보호 장구를 설치합니다.
- 전원이 켜져 있는 동안에는 구성 요소의 전기 연결 지점에 절대로 닿으면 안 됩니다.
- 구성 요소 전원이 켜져 있을 때에는 커넥터를 분리하거나 연결하지 마십시오.
- 특정 조건 하에서는 범용 전류에 민감한 잔류 전류로 작동하는 회로 차단기(RCD/RCM)가 보호하는 전원에서 전기 드라이브 시스템이 작동할 수 있습니다.
- 외부 하우징(예: 컨트롤 캐비닛)을 제공하여 이물질과 물이 들어가거나 직접 접촉하지 않도록 내부의 장치를 안전하게 보호합니다.

**고압의 하우징 전압과 고압의 누출 전류! 감전에 의한 사망이나 부상 위험!**

- 전원을 켜기 전과 가동하기 전에 전기 드라이브 및 컨트롤 시스템의 구성 요소를 장비 접지 단자의 접지 지점에 접지하거나 연결합니다.
- 전기 드라이브 및 컨트롤 시스템 구성 요소의 장비 접지 단자를 항상 주 전원 공급 장치에 연결합니다. 누출 전류는 3.5mA 보다 큼니다.
- 아래 표에 따라 각 최소 단면적을 가진 장비 접지 연결을 설정합니다. 외부 단자 단면적이 10 mm<sup>2</sup> (8 AWG) 이하인 경우에는 두 장비 접지 단자 연결도 허용되며, 이때 각 장비 접지 단자는 외부 단자와 동일한 단면적을 갖습니다.

| 외부 단자 단면적                    | 장비 접지 단자 최소 단면적<br>누출 전류 ≥ 3.5 mA |                                  |
|------------------------------|-----------------------------------|----------------------------------|
|                              | 1 장비 접지 단자                        | 2 장비 접지 단자                       |
| 1.5 mm <sup>2</sup> (AWG 16) | 10 mm <sup>2</sup> (AWG 8)        | 2 × 1.5 mm <sup>2</sup> (AWG 16) |
| 2.5 mm <sup>2</sup> (AWG 14) |                                   | 2 × 2.5 mm <sup>2</sup> (AWG 14) |
| 4 mm <sup>2</sup> (AWG 12)   |                                   | 2 × 4 mm <sup>2</sup> (AWG 12)   |
| 6 mm <sup>2</sup> (AWG 10)   |                                   | 2 × 6 mm <sup>2</sup> (AWG 10)   |
| 10 mm <sup>2</sup> (AWG 8)   |                                   | -                                |

전기 드라이브 및 컨트롤 안전 지침

| 외부 단자 단면적                    | 장비 접지 단자 최소 단면적<br>누출 전류 $\geq 3.5$ mA |            |
|------------------------------|----------------------------------------|------------|
|                              | 1 장비 접지 단자                             | 2 장비 접지 단자 |
| 16 mm <sup>2</sup> (AWG 6)   | 16 mm <sup>2</sup> (AWG 6)             | -          |
| 25 mm <sup>2</sup> (AWG 4)   |                                        | -          |
| 35 mm <sup>2</sup> (AWG 2)   |                                        | -          |
| 50 mm <sup>2</sup> (AWG 1/0) | 25 mm <sup>2</sup> (AWG 4)             | -          |
| 70 mm <sup>2</sup> (AWG 2/0) | 35 mm <sup>2</sup> (AWG 2)             | -          |
| ...                          | ...                                    | ...        |

표 3-1: 장비 접지 연결의 최소 단면적

### 3.3.2 감전 방지를 위한 매우 낮은 보호 전압

매우 낮은 보호 전압을 사용하면 전압이 매우 낮은 회로에 대해 기본적인 절연 기능을 지닌 장치를 연결할 수 있습니다.

Bosch Rexroth 에서 제공하는 전기 드라이브 및 컨트롤 시스템의 구성 요소에서 전압이 5V 와 50V 사이인 모든 연결 및 단자는 PELV("매우 낮은 보호 전압") 시스템입니다. 기본 절연 기능을 갖춘 장치(예: 프로그래밍 장치, PC, 노트북, 디스플레이 장치)를 이 연결 장치에 연결할 수 있습니다.

#### 감전에 의한 사망이나 부상 위험! 잘못된 연결에 의한 고전압!

50V 이상의 전압 및 회로가 포함된 장치(예: 메인 연결)의 전압이 매우 낮은 회로를 Bosch Rexroth 제품과 연결하는 경우 전압이 매우 낮은 연결 회로는 PELV("매우 낮은 보호 전압") 요구 사항을 충족해야 합니다.

### 3.3.3 위험한 기계 작동으로부터 보호

위험한 기계 작동은 연결 모터의 컨트롤 결함으로 인해 발생할 수 있습니다. 일반적인 예는 다음과 같습니다.

- 부적절하거나 잘못된 배선 또는 케이블 연결
- 운영자 오류
- 가동 전 잘못된 매개변수 입력
- 센서 및 엔코더 의 오작동
- 불량 구성 요소
- 소프트웨어 또는 펌웨어 오류

이러한 오류는 장비를 켜 직후나 문제 없이 지정되지 않은 시간 동안 작동한 뒤에도 발생할 수 있습니다.

일반적으로 전기 드라이브 및 컨트롤 시스템 구성 요소의 모니터링만으로도 연결된 드라이브에서의 오작동을 충분히 피할 수 있습니다. 하지만 개인의 안전, 특히 부상 및/또는 제품 손상의 위험과 관련하여 완벽하게 안전을 보장하려면 이것 하나만으로는 안 됩니다. 통합 모니터링 기능을 실제 사용할 수 있을 때까지는 언제든지 결함있는 드라이브 작동이 발생할 것이라고 가정해야 합니다. 결함있는 드라이브 작동 범위는 컨트롤의 유형 및 작동 상태에 따라 다릅니다.

**위험한 기계 작동! 생명 위험, 부상 위험, 심각한 부상 또는 제품 손상!**

## 전기 드라이브 및 컨트롤 안전 지침

전기 드라이브 및 컨트롤 시스템의 구성 요소가 설치되는 특정 조건에 따라 설치 또는 장비를 위한 **위험 점검**을 준비해야 합니다.

위험 점검의 결과로 사용자는 개인의 안전을 위해 설치 쪽에서 모니터링 기능 및 상위 레벨 조치를 마련해야 합니다. 설치 또는 장비에 적용되는 안전 규정을 고려해야 합니다. 의도하지 않은 장비 작동이나 기타 오작동은 안전 장치가 비활성화, 우회 또는 활성화되지 않을 경우 발생합니다.

**사고, 부상 및/또는 제품 손상을 방지하려면 다음 지침을 준수하십시오.**

- 장비의 작동 범위 및 작동부에 방해물을 치우고 안전하게 유지합니다. 실수로 사람들이 장비의 작동 범위 안으로 들어가지 못하게 하는 가능한 조치는 다음과 같습니다.
  - 안전 펜스
  - 안전 가드
  - 보호 장구
  - 조명 장벽
- 펜스와 보호 장구는 가능한 최대의 운동량에 충분히 저항할 수 있을 만큼 튼튼해야 합니다.
- 운영자가 급할 때 누를 수 있도록 비상 정지 스위치를 장착합니다. 가동하기 전에 비상 정지 장비가 제대로 작동하는지 확인합니다. 이때 비상 정지 스위치가 작동하지 않으면 장비를 작동하지 마십시오.
- 무심코 시작되지 않도록 OFF 스위치/OFF 버튼을 통해 드라이브 전원 연결을 해제하거나 안전 시작 잠금 기능을 사용합니다.
- 위험 지역에 접근하거나 출입하기 전에 드라이브가 안전하게 정지되었는지 확인합니다.
- 예를 들어 다음과 같이 모터 전원이 꺼진 다음 추락이나 낙하에 대비하여 수직축을 추가로 고정합니다.
  - 수직축을 기계 장치로 고정
  - 외부 브레이크/피뢰기/기계 잠금 장치 추가 또는
  - 수직축이 충분히 평행하도록 보장
- 표준 장비 **모터 정지 브레이크**나 드라이브 컨트롤러에서 제어하는 외부 정지 브레이크만으로는 **개인의 안전을 충분히 보장하지 못합니다!**
- 메인 스위치를 사용하여 전기 드라이브 및 컨트롤 시스템 구성 요소의 전원을 해제하고 다음과 같은 경우 다시 연결되지 않도록 구성 요소를 고정 ("잠금")합니다.
  - 정비 및 수리 작업
  - 장비 세척
  - 장비 사용을 오랜 기간 중단
- 전기 드라이브 및 컨트롤 시스템 구성 요소와 공급 리드 주위에서 고주파, 원격 컨트롤 및 무선 장비의 작동을 금지합니다. 이러한 장치를 불가피하게 사용해야 하는 경우 처음에 전기 드라이브 및 컨트롤 시스템을 가동하기 전에 가능한 모든 정상 사용 위치에서 이러한 고주파, 원격 컨트롤 및 무선 장비를 작동할 때 장비 또는 설치에 오작동이 발생하는지 확인합니다. 필요한 경우 특수 전자기 호환성(EMC) 테스트를 수행합니다.

### 3.3.4 작동 및 설치하는 동안 자기장 및 전자기장에 대비한 보호

전도체와 영구 자석에서 생성되는 자기장 및 전자기장은 심박 작동기, 금속 삽입물 및 보청기 착용자에게 심각한 위험을 의미합니다.

**전기 구성 요소 근처에서 심박 작동기, 금속 삽입물이나 보청기를 착용한 사람의 경우 건강상 위험!**

## 전기 드라이브 및 컨트롤 안전 지침

- 심박 작동기와 금속 삽입물을 사용하는 사람들은 다음과 같은 공간에 출입할 수 없습니다.
  - 전기 드라이브 및 컨트롤 시스템의 구성 요소가 설치, 가동 및 작동되는 공간
  - 영구 자석이 부착된 모터 부품이 보관, 수리 또는 설치되는 공간
- 심박 작동기를 부착한 사람이 그러한 공간에 반드시 출입해야 하는 경우 먼저 의사에게 문의해야 합니다. 삽입된 심박 작동기의 소음 면적은 기종마다 크게 다르므로 일반적인 규정이 적용될 수 없습니다.
- 금속 삽입물이나 금속 조각이 있는 사람들은 보청기 착용자와 마찬가지로 위에 명시된 공간에 출입하기 전에 먼저 의사와 상담해야 합니다.

### 3.3.5 고온 부품과의 접촉으로부터 보호

**전기 드라이브 및 컨트롤 시스템 구성 요소의 뜨거운 표면으로 인한 화상 위험!**

- 브레이크 저항, 히트싱크, 공급 장치, 드라이브 컨트롤, 모터, 권선 및 적층 코어 등과 같은 뜨거운 표면을 만지지 마십시오.
- 작동 조건에 따르면 작동 중이나 작동 직후 표면 온도가 **60°C(140°F)** 이상 올라갈 수 있습니다.
- 모터를 끈 다음 만지기 전에 오랜 시간 동안 충분히 모터를 식힙니다. 냉각 시간은 **최대 140 분이 소요될 수 있습니다!** 냉각에 필요한 대략적인 추정 시간은 기술 데이터에 명시된 온도 시정수의 5 배에 해당합니다.
- 초크, 공급 장치 및 드라이브 컨트롤러를 끈 다음 만지기 전에 **15 분** 간 냉각합니다.
- 안전 장갑을 착용하거나 고온 표면에서 작업하지 마십시오.
- 특정 적용의 경우 장비 또는 설치 제조업체가 각각의 안전 규정에 따라 최종 적용에서 화상으로 인한 부상이 발생하지 않도록 조치를 취해야 합니다. 이러한 조치의 예로는 장비 또는 설치에서의 경고, 가드(차폐 또는 장벽), 적용 설명서에 있는 안전 지침 등이 있습니다.

### 3.3.6 취급 및 설치하는 동안 보호

**부적절한 취급으로 인한 부상 위험! 타박상, 전단, 절개 및 타격으로 인한 신체 부상!**

- 사고 예방에 대한 관련 법적 규정을 준수하십시오.
- 설치 및 운반에 적합한 장비를 사용하십시오.
- 적절한 조치를 취하여 막힘과 분쇄를 방지하십시오.
- 항상 적합한 도구를 사용하십시오. 지정된 특수 도구를 사용하십시오.
- 리프트 장비 및 도구를 올바른 방법으로 사용하십시오.
- 적당한 보호 장비(예: 안전모, 안전 고글, 안전 작업화, 안전 장갑)를 사용하십시오.
- 머리 위에 걸려 있는 로드 밑에서 서 있지 마십시오.
- 바닥에 액체가 옆질러지면 미끄러져 넘어지지 않게 즉시 닦습니다.

### 3.3.7 배터리 안전

배터리는 단단한 하우징으로 둘러싸인 활성 화학 물질로 구성되어 있습니다. 따라서 잘못 취급하면 부상 또는 제품 손상이 발생할 수 있습니다.

**부적절한 취급으로 인한 부상 위험!**

## 전기 드라이브 및 컨트롤 안전 지침

- 가열이나 다른 방법으로 방전된 배터리를 재활용하지 마십시오(폭발 또는 부식 위험).
- 배터리를 재충전하면 누출 또는 폭발할 수 있으므로 재충전하지 마십시오.
- 배터리를 불 속으로 던지지 마십시오.
- 배터리를 분해하지 마십시오.
- 배터리 교체 시 장치에 설치된 전기 부품을 손상시키지 마십시오.
- 제품에 지정된 배터리 종류만 사용하십시오.



환경 보호 및 폐기! 법률 규정에 따르면 배터리가 제품 안에 들어있는 상태로 육상, 항공 및 해상 수송하는 경우 위험물(폭발 위험)로 간주됩니다. 사용한 배터리는 다른 쓰레기와 별도로 처분합니다. 해당 국가의 규정을 준수하십시오.

### 3.3.8 가압 시스템에 대한 보호

프로젝트 계획 설명서에 명시된 정보에 따라 액체 및 압축 공기로 냉각된 모터와 구성 요소에는 외부에서 공급되는 가압 매체(예: 압축 공기, 유압 오일, 냉각 윤활제)가 부분적으로 공급될 수 있습니다. 연결된 공급 시스템, 공급 라인 또는 연결 장치를 잘못 취급하면 부상 또는 제품 손상이 발생할 수 있습니다.

#### 가압 라인의 부적절한 취급으로 인해 생기는 부상 위험!

- 가압 라인을 연결 해제, 개봉 또는 절단하지 마십시오(폭발 위험).
- 각 제조업체의 작동 지침을 준수하십시오.
- 라인을 제거하기 전 압력을 낮춰 빈 매체로 만듭니다.
- 적당한 보호 장비(예: 안전 고글, 안전 작업화, 안전 장갑)를 사용하십시오.
- 바닥에 액체가 엇질러지면 미끄러져 넘어지지 않게 즉시 닦습니다.



환경 보호 및 폐기! 제품을 작동하는 데 사용되는 약품(예: 유체)은 환경 친화적이지 않을 수 있습니다. 환경에 유해한 약품은 다른 폐기물과 별도로 처분하십시오. 해당 국가의 규정을 준수하십시오.

## 3.4 신호 단어 및 안전 경고 기호에 대한 설명

사용 가능한 적용 설명서의 안전 지침에는 특정 신호 단어(DANGER, WARNING, CAUTION 또는 NOTICE) 및 필요한 경우 안전 경고 기호(ANSI Z535.6-2006 에 따름)가 포함되어 있습니다.

신호 단어는 안전 지침에 주의를 기울이게 하고 위험의 심각도를 나타냅니다.

신호 단어 DANGER, WARNING 및 CAUTION 앞에 오는 안전 경고 기호(느낌표가 있는 삼각형)는 신체 부상 위험을 경고하는 데 사용됩니다.



**위험**

이 안전 지침을 준수하지 않을 경우 사망 또는 심각한 부상이 발생합니다.

전기 드라이브 및 컨트롤 안전 지침

---

**⚠ 경고**

이 안전 지침을 준수하지 않을 경우 사망 또는 심각한 부상이 발생할 수 있습니다.

---

**⚠ 주의**

이 안전 지침을 준수하지 않을 경우 경미하거나 중간 정도의 부상이 발생할 수 있습니다.

---

**주의 사항**

이 안전 지침을 준수하지 않을 경우 제품 손상이 발생할 수 있습니다.

---



## 4 첫 단계 - IndraLogic

### 4.1 프로젝트 목표 "첫 단계 - IndraLogic"

간단한 신호등 컨트롤의 드래프트 및 구현을 통해 PLC 프로그래밍 및 컨트롤의 가동 방법을 소개합니다. PLC 프로그래밍 언어 ST(구조체언어) 및 시퀀셜 펄스 차트(AS), FBD(펄스블럭 다이어그램) 및 필요한 경우 CFC(연결구조언어)가 사용됩니다.

간단한 가동 시각화를 통해 프로젝트를 완료합니다.

이 섹션에서는 IndraWorks의 몇 가지 일반적인 작동 순서를 설명합니다. 이러한 설명에 따라 진행하면 IndraWorks의 작동 방식에 익숙해지고 각 구성 요소가 상호 작용하는 방식을 이해할 수 있습니다.

#### 프로젝트 복원:

사용하는 하드웨어(예: XLC L25, L45, L65(sercos III 버스 포함 또는 제외) 또는 MLC L25, L45, L65)에 관계없이 컨트롤을 사용하여 PC에서 프로젝트를 다시 생성할 수 있습니다.

이렇게 하려면 "Iw:\All Programs\Rexroth\IndraWorks\Projects" 아래의 설치 디렉터리에 있는 내보내기 파일 "Application\_FirstSteps\_IndraLogic\_Basics.iwx"를 사용합니다. 자세한 내용은 [프로젝트 복원, 78 페이지](#)를 참조하십시오.



기본적으로 IndraWorks는 "C:\Programs\Rexroth"에 설치됩니다. IndraWorks 설치 디렉터리는 지정된 설치 디렉터리 경로와 다를 수 있습니다.

#### 하위 작업

- 컨트롤을 사용하여 프로젝트 만들기
- PLC 프로그램 편집
- PLC 시각화 편집
- PLC 프로그램 시뮬레이션
- PLC 프로그램을 컨트롤로 로드

### 4.2 기술 요구 사항

- IndraMotion MLC용 IndraWorks Engineering이 설치된 PC, 12VRS, 컨트롤, 유형 "IndraMotion MLC L25", "IndraMotion MLC L45" 또는 "IndraMotion MLC L65", 최신 펌웨어가 로드되어 있어야 합니다.

이러한 사전 요구 사항 충족에 대해서는 다음 설명서를 참조하십시오.

- Rexroth IndraMotion MLC 12VRS 기능 설명, DOK-IM\*ML\*-SYSTEM\*\*V12-RExx-EN-P 및
- IndraWorks 온라인 도움말, **System IndraLogic XLC/IndraMotion MLC ▶ XLC/MLC 기능 설명(XLC/MLC Functional Description) ▶ 가동 및 작동(Commissioning and Operation)**,

컨트롤(예: 유형 "IndraLogic XLC L65.1"), 최신 펌웨어가 로드되어 있어야 합니다.

- IP 주소를 알아야 합니다.
- 컨트롤이 이더넷을 통해 프로그래밍 장치(PC)에 연결되어 있어야 합니다.
- sercos III 버스가 있어야 하며 브리지로 연결되어 있어야 합니다.

첫 단계 - IndraLogic

## 4.3 새 IndraWorks 프로젝트 만들기

### 4.3.1 컨트롤 재설정(CMLx5)

컨트롤에는 이전 프로그램 구성 요소, 매개변수 입력 및 PLC 데이터 등 이전 사용 기록이 남아 있을 수 있습니다. 이 단계에서는 메모리를 비워 이러한 데이터를 제거합니다.

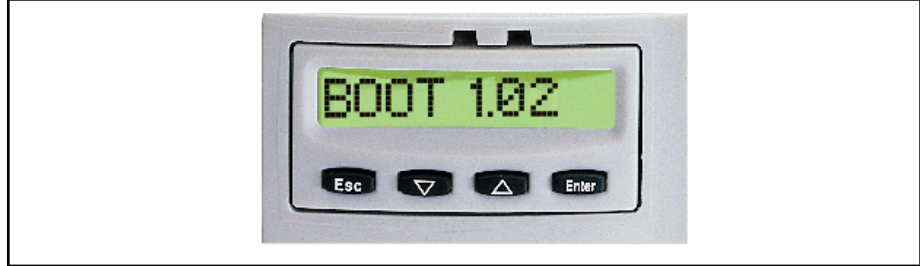


그림 4-1: 컨트롤 다시 부팅

컨트롤의 전원이 켜진 후에 디스플레이 아래에 있는 외부 버튼 <ESC>와 <Enter>를 누르고 부팅 1.02(BOOT 1.02)가 나타난 다음 부팅 중지(BOOTSTOP)가 나타날 때까지 버튼을 누르고 있습니다. 두 개의 내부 버튼을 누릅니다. 먼저 정리(CLEANUP)를 누른 다음 다시 부팅(REBOOT)을 누릅니다.

BB RDY/RDY 가 나타나고, 메모리를 비우고, 설치된 모션 구성 요소 및 PLC 구성 요소가 작동 준비될 때까지 컨트롤이 실행됩니다.

### 4.3.2 IndraWorks 시작

시작(Start) ▶ 프로그램(Programs) ▶ Rexroth ▶ IndraWorks ▶ Engineering 을 통해 IndraWorks 를 시작합니다.

IndraWorks Engineering 환경이 시작됩니다. 또는 바탕 화면 아이콘을 통해 IndraWorks 를 시작할 수도 있습니다.



바탕 화면 아이콘: 

### 4.3.3 새 프로젝트 만들기

새 프로젝트를 만들려면 작업 공간에서 "빈 프로젝트 만들기"(Create an empty project)를 클릭합니다.

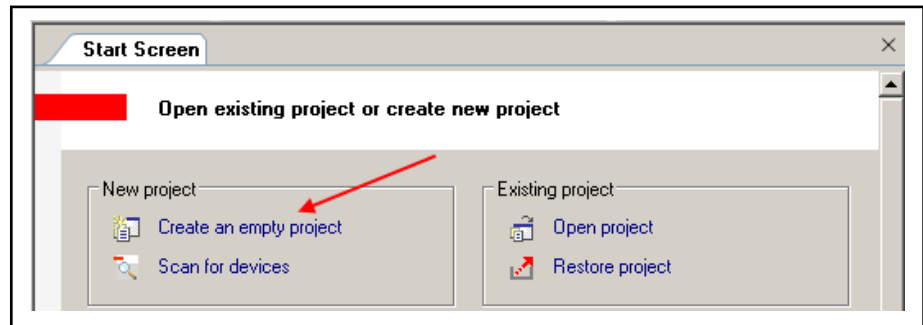


그림 4-2: 빈 프로젝트 만들기

또는 파일(File) ▶ 새로 만들기(New) ▶ 프로젝트...(Project...)를 통해 새 프로젝트를 만들 수도 있습니다.

새 IndraWorks 프로젝트의 프로젝트 이름 및 저장 위치 설정을 지정할 수 있는 마법사가 시작됩니다. 다음 그림을 참조하십시오.

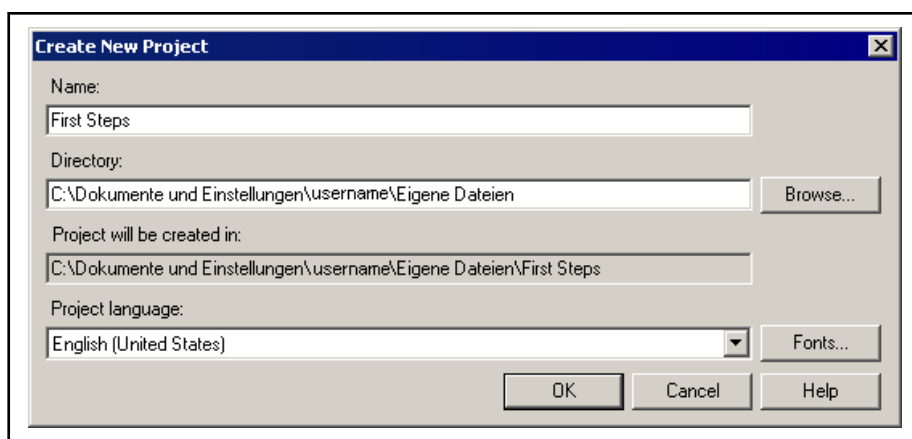


그림 4-3: 새 IndraWorks 프로젝트를 저장하는 설정

- 이름(Name)** 프로젝트의 이름을 지정합니다. 제안: "첫 단계"(First steps).
- 디렉터리(Directory)** 이 필드에는 기본적으로 C:\Documents and Settings\username\MyDocuments 디렉터리의 저장 위치가 포함되어 있습니다. 다른 디렉터리를 지정할 수도 있습니다.
- 프로젝트를 만들 위치(Project is created in)** 디렉터리 및 프로젝트 이름이 자동으로 생성됩니다.
- 프로젝트 언어(Project language)** 프로젝트 언어를 선택하면 사용자 텍스트의 언어가 결정됩니다. 프로젝트 탐색기에 이전에 정의한 프로젝트 이름을 사용하여 새로 만든 프로젝트가 나타납니다. 다음 그림을 참조하십시오.

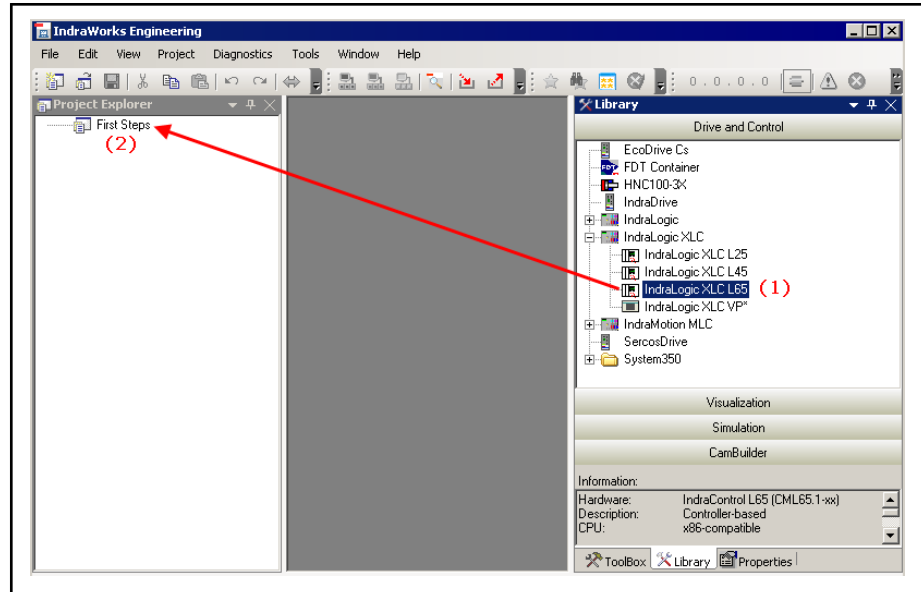


그림 4-4: 첫 단계 프로젝트

## 4.4 프로젝트에 새 제품 추가

"드라이브 및 컨트롤"(Drive and Control) 라이브러리에서 원하는 제품(예: "IndraLogic XLC", "IndraLogic XLC L65" 폴더)를 프로젝트 탐색기의 새로 만든 프로젝트로 끌어다 놓습니다.

## 첫 단계 - IndraLogic



- (1) 제품: IndraLogic XLC L65  
 (2) 프로젝트: 첫 단계(First Steps)

그림 4-5: 제품 추가

그러면 아래 설명된 대로 필요한 기본 설정을 지정할 수 있는 마법사가 자동으로 시작됩니다.

## 4.5 제품에 기본 설정 할당

### 4.5.1 제품설정

추가할 제품에 대한 일반적인 정보를 입력합니다.

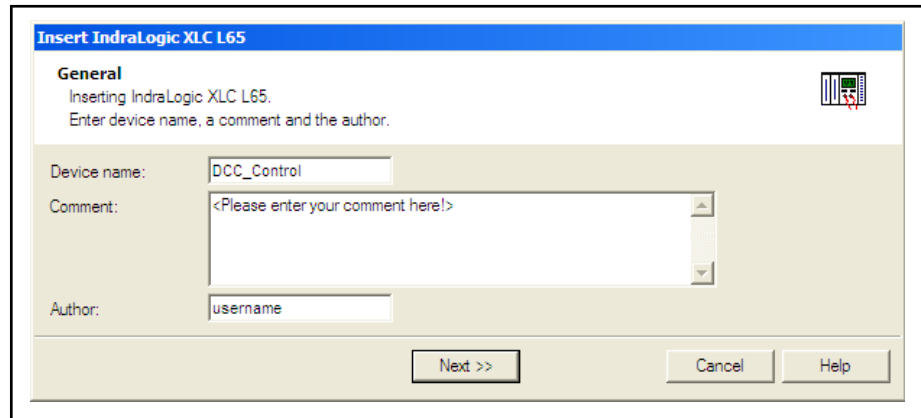


그림 4-6: 제품설정

- 제품 이름(Device name)** "제품 이름"(Device Name)에는 원하는 제품 이름(예: "DCC\_Control")을 입력합니다. 기본적으로 이 필드에는 라이브러리에 표시된 이름이 포함되어 있습니다. 나중에 제품은 프로젝트 탐색기에 지정된 이름으로 나타납니다.
- 설명(Comment)** 제품에 대한 자세한 설명을 추가합니다. 이 필드는 선택 사항입니다.
- 작성자(Author)** 프로젝트 작성자의 이름을 입력합니다. 이 필드는 선택 사항입니다.



이러한 입력 사항은 나중에 변경할 수도 있습니다.

"다음>>"(Next>>)을 클릭하여 입력 사항을 확인합니다.

## 4.5.2 통신 설정

"구성" 대화 상자에서 펌웨어 및 IndraWorks 와 제품(이 경우 IndraLogic XLC L65) 간 통신에 대한 설정을 정의합니다.

그림 4-7: 구성

|                          |                                                                                                                                                                                                                                                                     |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 제품유형(Device type)        | 제품유형은 컨트롤 선택을 통해 미리 결정됩니다. 하나의 제품유형에 대해 여러 가지 다른 구성이 있는 경우 수동 조정이 필요합니다. 예를 들어, IndraLogic XLC Lx5 는 sercos III 버스와 함께 사용하거나(CMLx5.1-3P), sercos III 버스 없이 사용할 수 있습니다(CMLx5.1-NP).                                                                                  |
| 펌웨어 버전(Firmware version) | 여기에서는 펌웨어 버전을 선택합니다. 일반적으로 이 설정은 유지될 수 있습니다.                                                                                                                                                                                                                        |
| 펌웨어 출시(Firmware release) | 여기에서는 펌웨어 버전 내의 펌웨어 출시를 설정합니다. 이 설정도 유지되어야 합니다.                                                                                                                                                                                                                     |
| IP 주소(IP address)        |  이더넷 통신 항목은 <b>시뮬레이션 모드</b> , 54 페이지에서는 평가되지 않습니다. 연결을 테스트할 필요가 없습니다.                                                                                                            |
| PLC 게이트웨이(PLC Gateway)   | "제품 IP 주소"에 대상 주소(즉, 응용 프로그램을 로드해야 하는 컨트롤의 IP 주소)를 입력합니다.<br>"게이트웨이 IP 주소"에서 게이트웨이 서버가 실행 중인 컴퓨터의 주소를 정의합니다. 여기에서 기본 컴퓨터 이름(주소)은 "로컬 호스트"(localhost)입니다. 즉, 로컬로 설치된 게이트웨이의 주소를 지정합니다. 이 설정은 변경되지 않습니다.                                                              |
| 연결 테스트(Connection test)  | "연결 테스트"(Connection Test) 버튼을 클릭하여 컴퓨터와 대상 컨트롤 간 통신이 성공적으로 수행되는지 확인합니다. "연결 테스트 결과"(Connection test result) 필드에 테스트 결과가 기록됩니다.<br>PLC 통신 테스트가 성공적으로 수행된 경우 IP 주소가 표시됩니다.<br>발생할 가능성이 있는 모든 문제는 "연결 테스트 결과"(Connection Test Result) 창에 빨간색 또는 노란색의 깜박이는 아이콘으로 표시됩니다. |

## 첫 단계 - IndraLogic

다음과 같은 문제가 발생할 수 있습니다.

- 출력: "컨트롤과의 통신에 실패했습니다."

원인:

- 컨트롤의 전원이 켜져 있지 않습니다.
- 네트워크 케이블이 연결되어 있지 않습니다.
- 컨트롤의 IP 주소와 설정된 IP 주소가 일치하지 않습니다.

해결 방안: 대화 상자에서 IP 주소를 변경하거나 컨트롤의 IP 주소를 변경합니다. Rexroth IndraMotion MLC 기능 설명의 컨트롤 가동을 참조하십시오.

- PC 및/또는 컨트롤의 네트워크 마스크가 너무 작게 설정되었습니다.

해결 방안: 네트워크 설정을 확인하고 수정합니다. Rexroth IndraMotion MLC 기능 설명의 컨트롤 가동을 참조하십시오.

- 출력: "구성된 펌웨어와 컨트롤의 펌웨어가 일치하지 않습니다."

원인:

- 컨트롤 펌웨어가 최신 버전이 아닙니다.

해결 방안: 펌웨어를 업데이트합니다. Rexroth IndraMotion MLC 기능 설명의 컨트롤 가동을 참조하십시오.

- 선택한 컨트롤이 실제 컨트롤 유형과 일치하지 않습니다.

안전한 온라인 모드(Secure online mode)

PLC 프로그래밍 언어(PLC programming language)

이 옵션을 선택하면 보안 관련 작업을 실행하기 전에 확인하는 프롬프트가 나타납니다.

여기에서 기본 PLC 프로그래밍 언어를 설정합니다. ST(구조체언어)를 선택합니다.

"다음 >>"(Next >>)을 클릭하여 다음 페이지로 이동합니다.

## 4.5.3 평선모듈설정



제품유형에 따라 "제품 및 평선모듈설정" 대화 상자가 달라질 수 있습니다. 이 프로젝트에서는 "IndraLogic XLC L65" 제품의 대화 상자를 예로 들어 설명합니다.

원하는 제품 및 평선모듈설정을 입력합니다.

**Insert IndraLogic XLC L65**

**Interfaces**  
Please select which onboard components and function modules you want to use.

SERCOSIII configuration (X7E1/X7E2): SercosIII Master

Slot0 Configuration (X7P): Profibus DP Master

Slot1 Configuration (X7E3/X7E4): Nicht verwendet

Ethernet configuration (X7E5): Not used

Function modules: Not used

Not used

Not used

Not used

그림 4-8: 평선모듈설정

첫 단계 - IndraLogic

|                                         |                                                                                                                                                           |
|-----------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| sercos III 구성(sercos III configuration) | 제품을 추가할 때 선택할 수 있는 현재 sercos III 구성입니다. "sercos III 마스터"(sercos III Master) 또는 "사용하지 않음"(Not used) 중에서 선택할 수 있습니다.                                        |
| 슬롯 0 구성(Slot0 configuration)            | 제품을 추가할 때 선택할 수 있는 현재 PROFIBUS 구성입니다. "PROFIBUS DP 마스터"(PROFIBUS DP Master), "PROFIBUS DP 슬레이브"(PROFIBUS DP Slave) 또는 "사용하지 않음"(Not Used) 중에서 선택할 수 있습니다. |
| 슬롯 1 구성(Slot1 configuration)            | 제품을 추가할 때 선택할 수 있는 현재 PROFINET 또는 이더넷/IP 구성입니다. "사용하지 않음"(Not Used), PROFINET 또는 이더넷/IP 인터페이스 중에서 선택할 수 있습니다.                                             |
| 이더넷 구성(EtherNet configuration)          | 이 인터페이스는 Engineering 인터페이스로 필요하므로(PC에 직접 연결 또는 작동 네트워크를 통해 연결) 이더넷/IP 어댑터가 사용되지 않을 수 있습니다.                                                                |
| 평선모듈(Function Modules) 1-4              | 평선모듈은 최대 4 개까지 "IndraLogic XLC L65"에 연결할 수 있습니다.<br>필요한 경우 각 목록에서 선택할 수 있습니다.<br>"완료"(Finish)를 클릭하여 입력 사항을 확인합니다.<br>프로젝트 탐색기에 새로 만든 제품이 나타납니다.           |

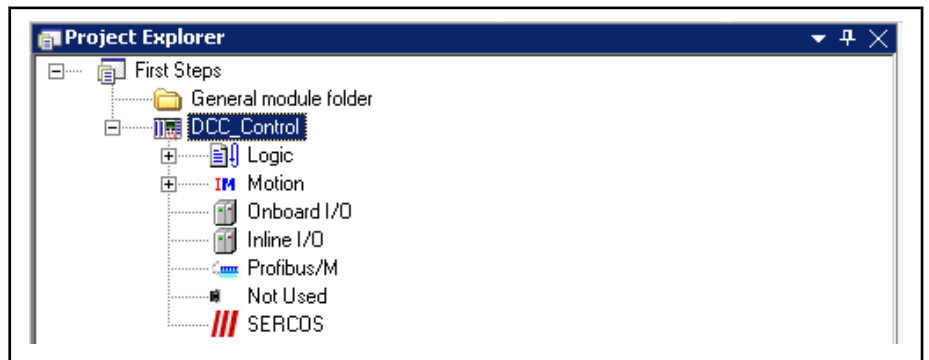


그림 4-9: 프로젝트 탐색기: sercos III 버스를 포함하는 IndraLogic XLC L65 컨트롤이 있는 "첫 단계" 프로젝트



프로젝트 탐색기의 화면에는 "구성"(Configuration) 및 "평선모듈"(Function Modules) 대화 상자에서 선택한 구성 요소가 반영됩니다.

## 4.6 PLC 프로그램 만들기

### 4.6.1 신호등 시스템의 컨트롤

아래에서는 예제 프로그램 "미니 교통 신호"에 대해 설명합니다. 이 프로그램은 교차로에서 두 개의 신호등을 제어합니다. 두 개의 신호등 모두 빨간색과 녹색 단계 사이에서 전환합니다. 사고를 방지하기 위해 빨간색과 녹색 단계 사이에 노란색 전환 단계 또는 노란색 및 빨간색 전환 단계를 추가해야 합니다.

이 예제에서는 다음을 확인할 수 있습니다.

1. IEC 61131-3 언어 구성을 사용하여 시간 종속 프로그램이 표시되는 방법
2. IndraLogic 을 사용하여 표준의 다양한 언어를 편집하는 방법
3. 프로그램 일부를 서로 연결할 수 있는 방법

### 4.6.2 프로그램 및 평선블럭 만들기

#### "CFC(연결구조언어)" 또는 "ST(구조체언어)"에서 "PlcProg" 프로그램 추가

POU(프로그램 유닛) "MlcProg" 및 "PlcProg"는 "ST(구조체언어)" 언어를 사용하여 미리 설정되며 L65 를 만든 후에 바로 사용할 수 있습니다.

"MlcProg"는 이 프로젝트에 필요하지 않지만 방해가 되지는 않습니다.

## 첫 단계 - IndraLogic



"PlcProg" 프로그램의 경우 다음 중에서 선택할 수 있습니다.

- "ST(구조체언어)"에서 만들 경우 이 장의 나머지 부분을 생략하고 AS에서 SEQUENCE 프로그램 추가(31 페이지)의 내용을 진행합니다.
- "CFC(연결구조언어)"(Continuous Function Chart (CFC))로 만들 경우에는 아래로 계속 진행합니다.

다음과 같이 진행합니다.

1. 프로젝트 탐색기에서 "PlcProg" POU(프로그램 유닛)를 강조합니다.
2. 마우스 오른쪽 버튼을 클릭하고 상황에 맞는 메뉴에서 삭제(Delete) 메뉴 항목을 선택합니다.



강조된 "PlcProg" POU(프로그램 유닛)는 <Del> 키를 사용하여 삭제할 수도 있습니다.

이제 "PLC 개체/일반 개체"(PLC Objects / General Objects) 라이브러리에서 "응용 프로그램"(Application) 노드로 POU(프로그램 유닛)를 끌어다 놓아 프로젝트에 새 POU(프로그램 유닛)를 추가합니다. 또는 응용 프로그램 노드의 상황에 맞는 메뉴에서 추가(Add) ▶ POU(프로그램 유닛) 명령을 사용할 수 있습니다.

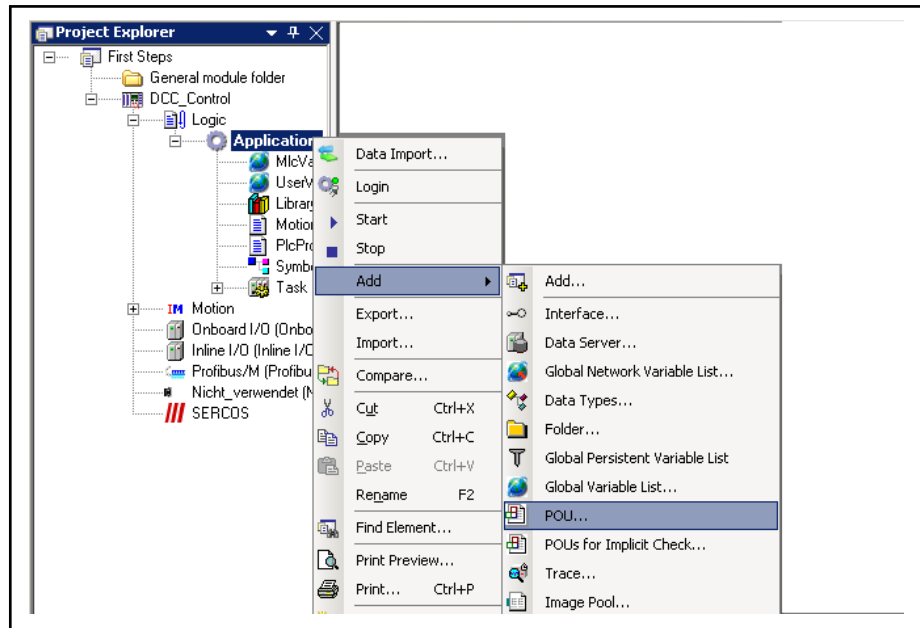


그림 4-10: 라이브러리에서 POU(프로그램 유닛) 추가

"개체 추가"(Add Object) 대화 상자가 열립니다. "개체 추가"(Add Object) 대화 상자에 다음 설정을 입력합니다.

1. "이름"(Name) 입력 필드에서 POU(프로그램 유닛) 이름을 "PlcProg"로 지정합니다.
2. "유형:"(Type:) 아래의 선택 영역에서 "프로그램"(Program)을 선택하고 "실행언어:"(Implementation language:)(프로그래밍 언어)로 "CFC(연결 구조언어)"(Continuous Function Chart (CFC))를 선택합니다.  
"완료"(Finish)를 사용하여 확인합니다.

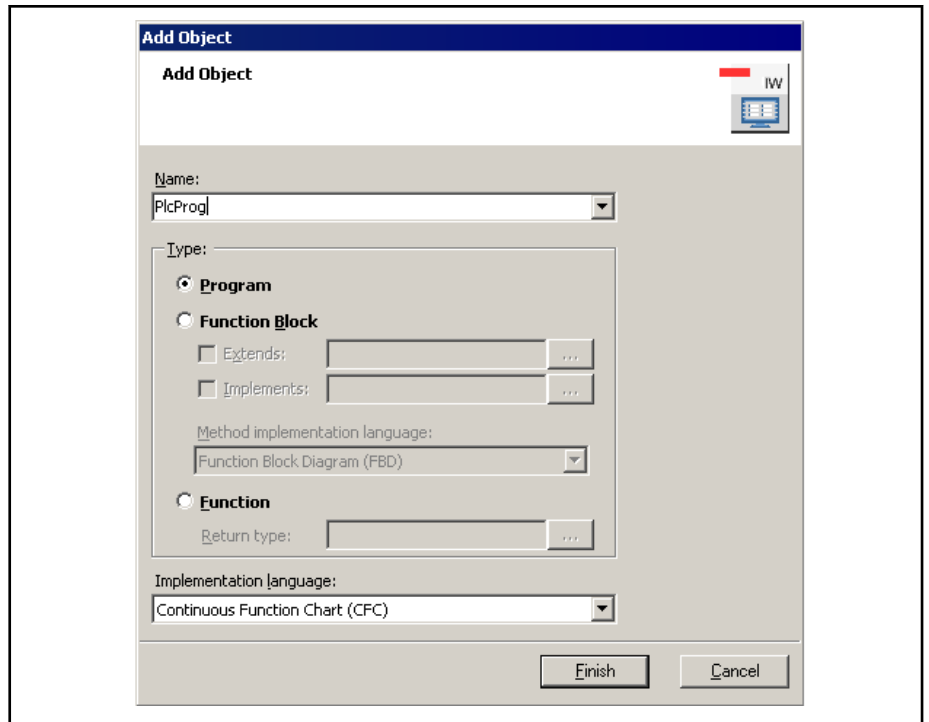


그림 4-11: "PlcProg" POU(프로그램 유닛) 구성

이제 프로젝트에서 "CFC(연결구조언어)" 언어로 된 POU(프로그램 유닛) 개체 "PlcProg"를 사용할 수 있습니다.

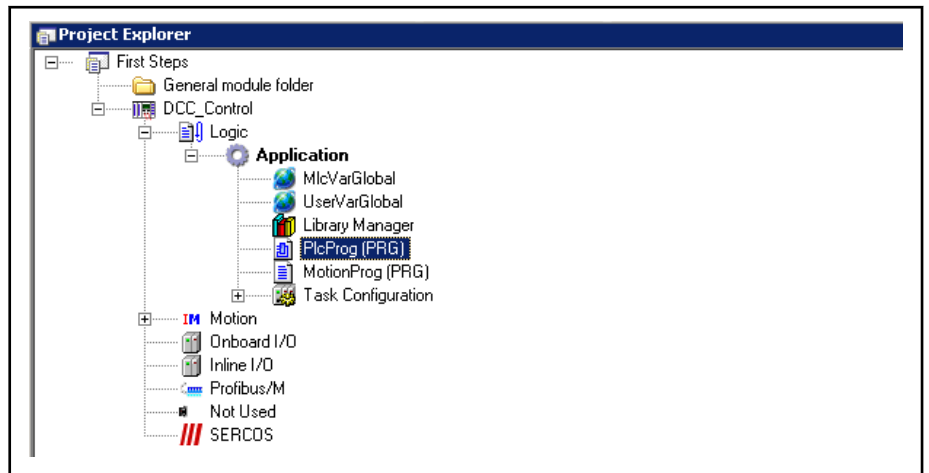


그림 4-12: "PlcProg" POU(프로그램 유닛)가 응용 프로그램에 추가되었습니다.

"PlcProg"의 기능 "PlcProg"에서 들어오는 시작 신호는 신호 단계 시퀀스와 결합되고 각 교통 신호의 색을 제어하기 위한 출력이 작동합니다.

## "AS"에서 "SEQUENCE" 프로그램 추가

"응용 프로그램"(Application)을 강조하고 마우스 오른쪽 버튼을 클릭합니다. 상황에 맞는 메뉴에서 **추가(Add) ▶ POU(프로그램 유닛)**를 선택합니다.

"개체 추가"(Add Object) 대화 상자가 열립니다. "개체 추가"(Add Object) 대화 상자에 다음 설정을 입력합니다.

1. "이름"(Name) 입력 필드에서 POU(프로그램 유닛) 이름을 "SEQUENCE"로 지정합니다.

## 첫 단계 - IndraLogic

2. "유형:"(Type:) 아래의 선택 영역에서 "프로그램"(Program)을 선택하고 "실행언어:"(Implementation language:)(프로그래밍 언어)로 "SFC(시퀀셜 펄스 차트)"(Sequential Function Chart (SFC))를 선택합니다.
3. "완료"(Finish)를 사용하여 확인합니다.

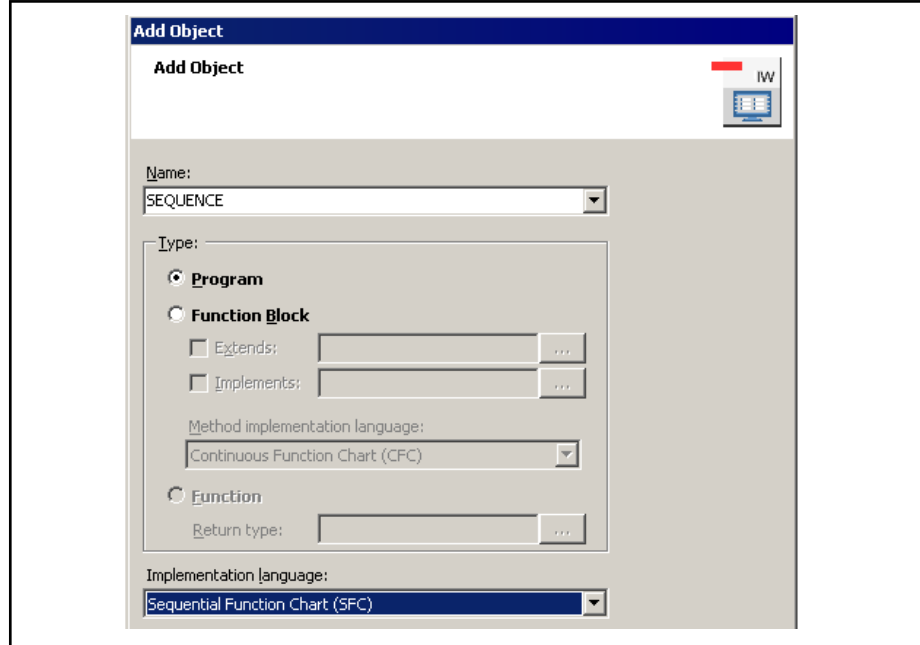


그림 4-13: "SEQUENCE" 프로그램 추가

이제 프로젝트에서 "SEQUENCE" 개체 프로그램을 사용할 수 있습니다.

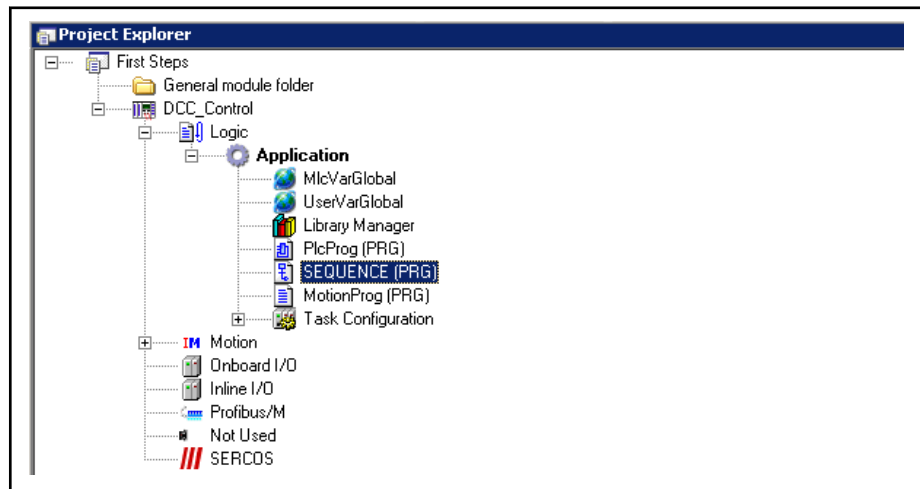


그림 4-14: 응용 프로그램의 일부가 된 "SEQUENCE"

**"SEQUENCE"의 기능**

"SEQUENCE" 프로그램에서는 교통 신호의 컨트롤 단계의 시퀀스가 프로그래밍됩니다. 예를 들어, 올바른 신호가 올바른 때에 올바른 시간 동안 켜지도록 "WAIT" FB(펄스블럭)가 호출되거나 각 신호 단계의 값이 제품의 "TRAFFICSIGNAL"에 저장됩니다.

**"FBD(펄스블럭 다이어그램)"에서 "TRAFFIC SIGNAL" 프로그램 추가**

"응용 프로그램"(Application)을 강조하고 마우스 오른쪽 버튼을 클릭합니다. 상황에 맞는 메뉴에서 **추가(Add) > POU(프로그램 유닛)**를 선택합니다.

"개체 추가"(Add Object) 대화 상자가 열립니다. "개체 추가"(Add Object) 대화 상자에 다음 설정을 입력합니다.

첫 단계 - IndraLogic

1. "이름"(Name) 입력 필드에서 POU(프로그램 유닛) 이름을 "TRAFFICSIGNAL"로 지정합니다.
2. "유형:"(Type:) 아래의 선택 영역에서 "평선블럭"(Function Block)을 선택하고 "실행언어:"(Implementation language:)(프로그래밍 언어)로 "FBD(평선블럭다이어그램)"(Function Block Diagram (FBD))을 선택합니다.
3. "완료"(Finish)를 사용하여 확인합니다.

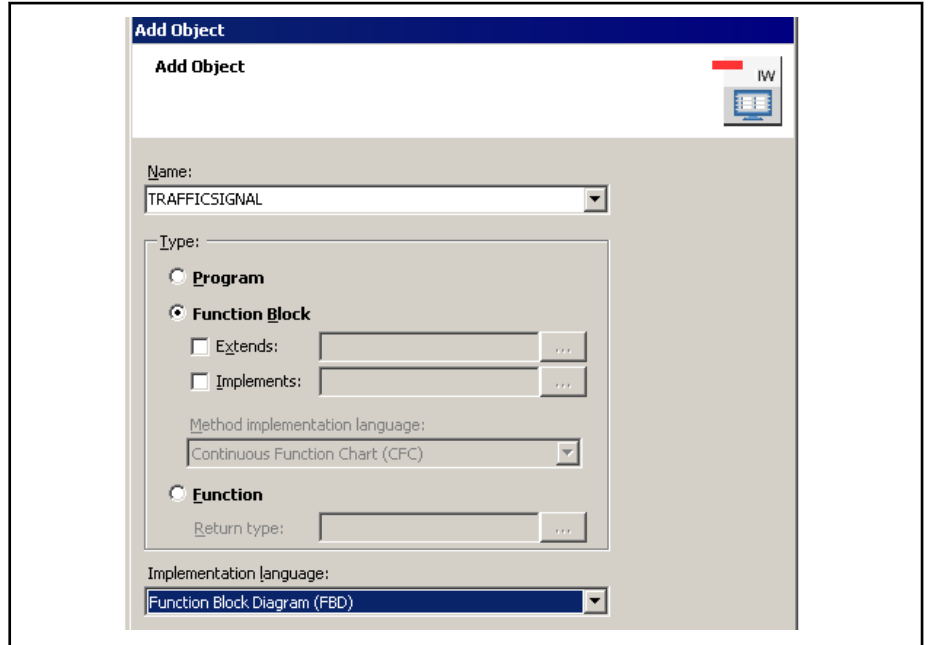


그림 4-15: "TRAFFICSIGNAL" 평선블럭 추가

이제 프로젝트 트리의 응용 프로그램 아래에 "평선블럭" 유형의 "TRAFFICSIGNAL" POU(프로그램 유닛)가 나타납니다.

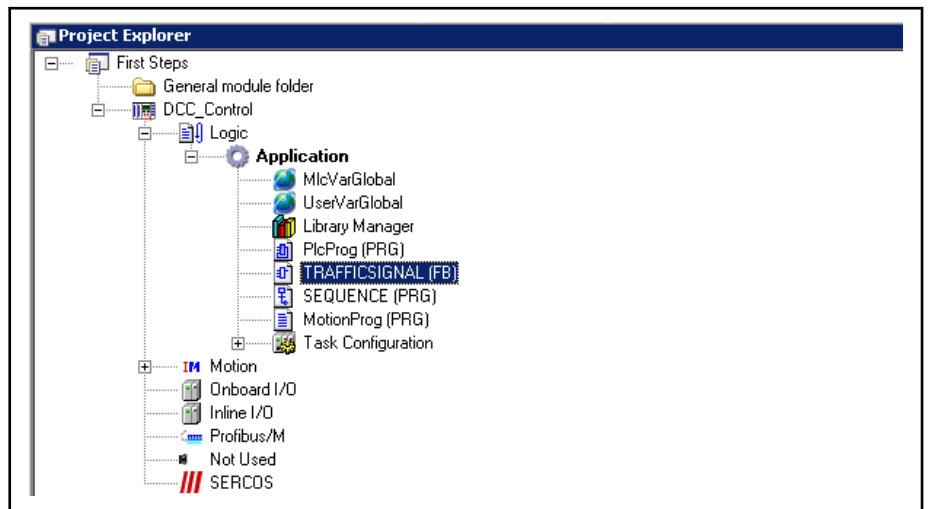


그림 4-16: 응용 프로그램의 일부가 된 "TRAFFICSIGNAL" 평선블럭

"TRAFFICSIGNAL"의 기능

"TRAFFICSIGNAL" 평선블럭에서는 신호등에 각 신호 단계가 할당됩니다. 빨간색 또는 노랑-빨간 신호 단계 중에는 빨간색 신호등이 작동합니다. 노란색 또는 노랑-빨간 신호 단계 중에는 노란색 신호등이 작동합니다. 녹색 신호 단계 중에는 녹색 신호등이 작동합니다. 다음 표를 참조하십시오.

## 첫 단계 - IndraLogic

| 신호 단계   | 빨간색 신호등 켜짐 | 노란색 신호등 켜짐 | 녹색 신호등 켜짐 |
|---------|------------|------------|-----------|
| 빨간색     | X          |            |           |
| 노란색-빨간색 | X          | X          |           |
| 노란색     |            | X          |           |
| 녹색      |            |            | X         |

표 4-1: 신호 단계 할당

**"ST(구조체언어)"에서 평선블럭 "WAIT" 추가**

마지막으로 추가할 FB(평선블럭)는 "WAIT"라는 POU(프로그램 유닛)입니다.

"응용 프로그램"(Application)을 강조하고 마우스 오른쪽 버튼을 클릭합니다. 상황에 맞는 메뉴에서 **추가(Add) ▶ POU(프로그램 유닛)**를 선택합니다.

"개체 추가"(Add Object) 대화 상자가 열립니다. "개체 추가"(Add Object) 대화 상자에 다음 설정을 입력합니다.

1. "이름"(Name) 입력 필드에서 POU(프로그램 유닛) 이름을 "WAIT"로 지정합니다.
2. "유형:"(Type:) 아래의 선택 영역에서 "평선블럭"(Function Block)을 선택하고 "실행언어:"(Implementation language:)(프로그래밍 언어)로 "ST(구조체언어)"(Structured Text (ST))를 선택합니다.
3. "완료"(Finish)를 사용하여 확인합니다.

다음 그림을 참조하십시오.

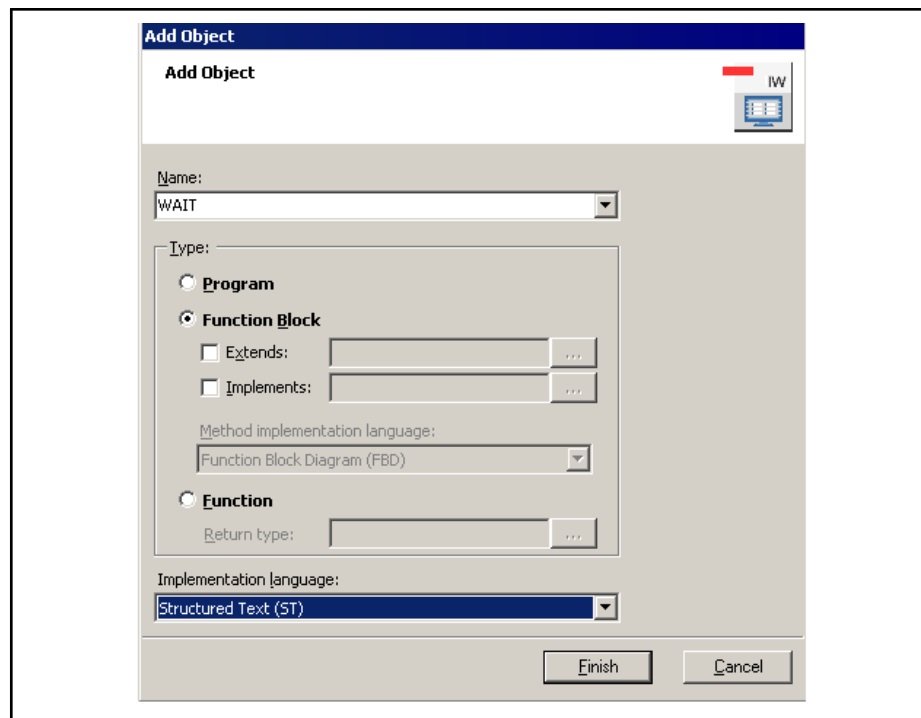


그림 4-17: "WAIT" 평선블럭 추가

이제 "평선블럭" 유형의 "WAIT" POU(프로그램 유닛)가 응용 프로그램의 일부가 되었습니다.

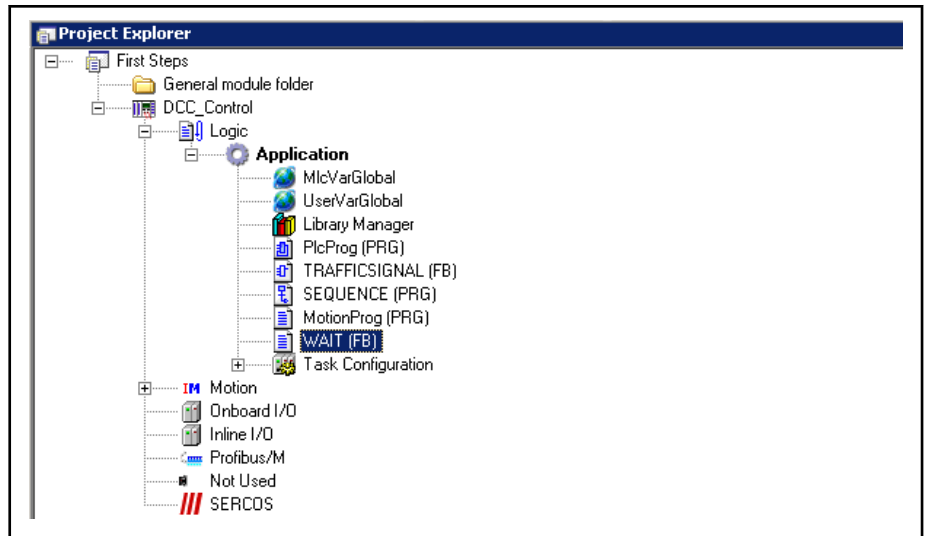


그림 4-18: 응용 프로그램의 일부가 된 "WAIT" 평선블럭

#### "WAIT"의 기능

"WAIT" POU(프로그램 유닛) 평선블럭에서는 IndraWorks 표준 라이브러리의 "TP" 평선블럭을 사용하는 간단한 시간 단계가 프로그래밍됩니다. 이 타이머를 사용하여 각 신호 단계의 길이를 입력할 수 있습니다. FB(평선블럭)의 입력 변수는 신호 단계의 기간(밀리초)이고 기간이 만료되는 즉시 부울 값 "TRUE"를 출력 변수로 전달합니다.

### 4.6.3 프로그램 및 평선블럭 편집 - 1 부

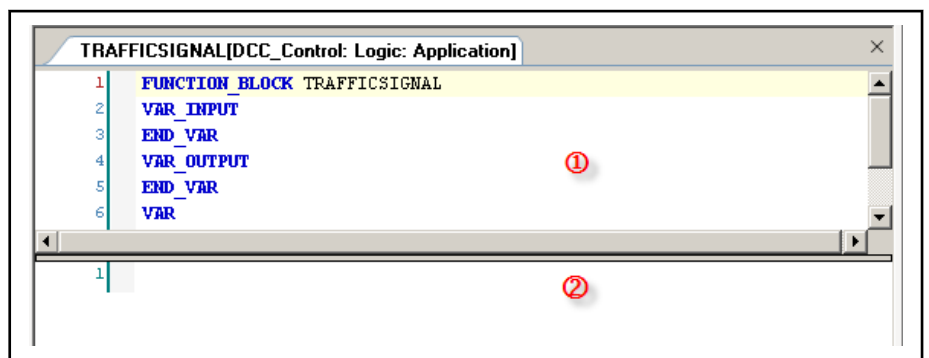
#### "FBD(평선블럭 다이어그램)"에서 "TRAFFIC SIGNAL" 평선블럭 편집

##### POU(프로그램 유닛) 열기

커서를 "TRAFFICSIGNAL" POU(프로그램 유닛) 위에 두고 마우스 오른쪽 버튼을 클릭하여 상황에 맞는 메뉴를 열어 "TRAFFICSIGNAL" 평선블럭을 엽니다. 상황에 맞는 메뉴에서 **열기(Open)**를 선택합니다. "TRAFFICSIGNAL" 평선블럭의 편집기가 열립니다. 다음 그림 "'TRAFFICSIGNAL' 평선블럭의 편집기가 열림"을 참조하십시오.



또는 POU(프로그램 유닛)를 두 번 클릭하여 편집기를 열 수도 있습니다.



- (1) 선언
- (2) 구현 섹션 또는 본문

그림 4-19: "TRAFFICSIGNAL" 평선블럭의 편집기 열기

##### POU(프로그램 유닛) 편집기

편집기는 다음 두 개의 섹션으로 분리되어 있습니다.

1. 선언
2. 구현 섹션

선언영역에서는 평선블럭의 입력 및 출력 변수와 로컬 변수가 생성됩니다.

## 첫 단계 - IndraLogic

구현 섹션에서는 사용자 프로그램을 편집합니다.

**변수 선언** POU(프로그램 유닛)에서 변수를 선언하는 방법에는 다음 두 가지가 있습니다.

1. 선언에 직접 변수를 배치합니다.
2. 선언 없이 프로그래밍을 시작한 다음 시스템에서 식별한 데이터를 입력 창에서 추가합니다(권장).

프로그램에서 입력한 변수를 인식하지 못하면 입력 창이 자동으로 열립니다.



1. 선언영역에서 직접 구조(STRUCT) 및 데이터 배열(ARRAY)의 평선블럭의 인스턴스를 만듭니다.
2. 프로그래밍 중에 입력 창을 사용하여 변수를 더 간단하게 만들 수 있습니다.  
유효 범위 및 올바른 데이터 유형을 올바르게 할당하십시오.
3. "선언된" 변수가 다시 제공될 경우 일반적으로 이는 입력 오류입니다.

**사용자 프로그램 만들기**

"TRAFFICSIGNAL" 평선블럭은 "STATUS", "INTEGER" 유형을 입력 변수로 받아 프로그램을 처리합니다. 해당하는 신호 단계에 대한 출력 변수 "GREEN", "YELLOW" 및 "RED"는 "TRUE"로 설정됩니다. "TRAFFICSIGNAL" 평선블럭은 나중에 "PlcProg"에서 호출됩니다.

**첫 번째 네트워크 편집**

명령문 섹션에서 사용자 프로그램을 편집합니다. 편집기에서 "1" 옆을 클릭하여 명령문 섹션에서 첫 번째 네트워크를 선택합니다. 마우스 오른쪽 버튼을 사용하여 상황에 맞는 메뉴를 열고 **FB(평선블럭) 호출 삽입(Insert FB call)**을 선택합니다. 또는 메뉴에서 **FBD/LD/IL ▶ FB(평선블럭) 호출 삽입(Insert FB call)**을 선택합니다.

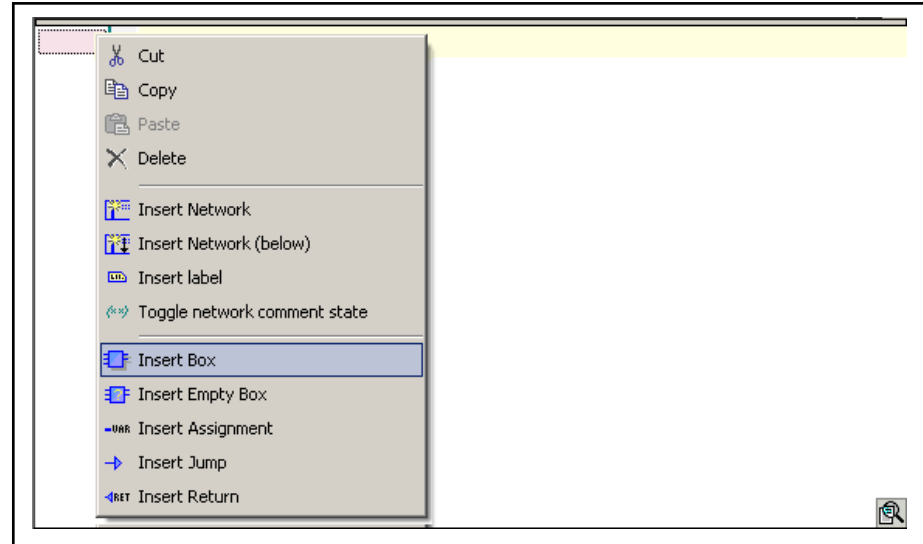


그림 4-20: FB(평선블럭) 호출 추가

"입력 보조툴"(Input Assistance) 창이 열립니다.

첫 단계 - IndraLogic

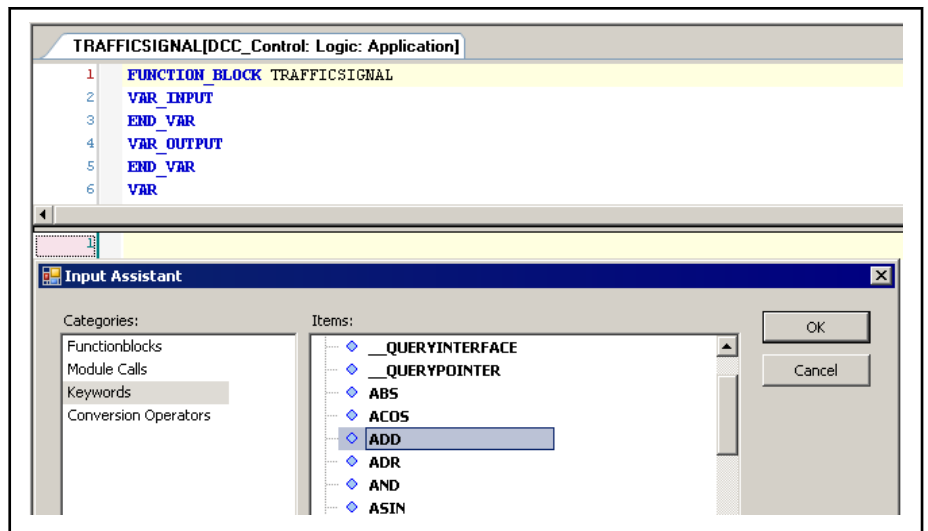


그림 4-21: 입력 보조툴 - 평선블럭 "ADD" 추가

첫 번째 네트워크에서는 연산자 "ADD"와 두 개의 입력이 포함된 상자가 생성됩니다.

1. "입력 보조툴"(Input Assistance)의 "범주:"(Categories:) 아래에서 "키워드"(Keywords)를 선택합니다.
2. "요소:"(Elements:) 아래에서 연산자 "ADD"를 강조합니다.  
"확인"(OK)을 클릭하여 확인합니다.

명령문 섹션이 다음과 같이 표시됩니다.

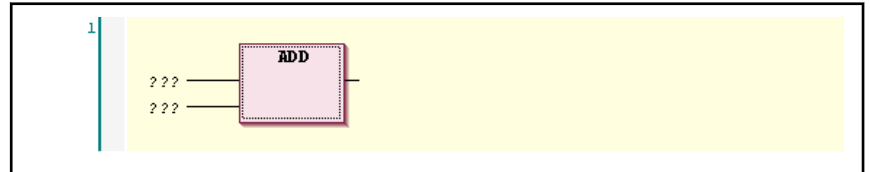


그림 4-22: 네트워크에 추가된 "ADD" 평선블럭

이제 "ADD" 평선블럭에 다른 이름과 기능을 지정합니다.

1. "ADD" 텍스트 위에 커서를 놓고 클릭하여 텍스트를 강조한 다음 "EQ"로 변경합니다.
2. "EQ" 왼쪽의 평선블럭에서 위쪽 "???" 입력을 강조하고 변수 이름으로 "STATUS"를 입력합니다.  
변수 선언 창이 열립니다.  
VAR, STATUS, INT 가 제공됩니다.
3. "STATUS"는 평선블럭의 입력 변수이므로 제안은 "VAR\_INPUT, STATUS, INT"로 변경되어야 합니다.

## 첫 단계 - IndraLogic

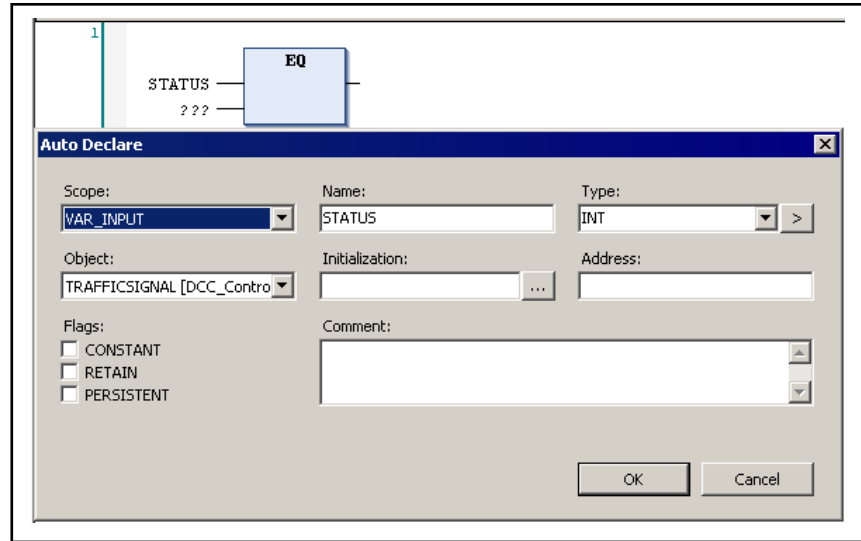


그림 4-23: "STATUS" 입력 변수 선언

평선블럭의 선언영역에 있는 "VAR\_INPUT...END\_VAR" 영역에 선언 줄 STATUS: INT;가 나타납니다.

- 아래쪽 "???" 입력에 "1"을 입력합니다.

"STATUS"는 서로 다른 신호 단계에 대해 네 가지 가능한 상태를 갖습니다. 이러한 상태는 "녹색", "노란색", "노란색-빨간색" 및 "빨간색"입니다.

네 가지 상태는 출력 변수로 디코딩해야 합니다. 네 가지 상태에는 "GREEN", "YELLOW" 및 "RED", 이렇게 세 가지 변수가 필요합니다.

- 이제 네트워크 "1"에서 "EQ" 평선블럭 다음의 영역을 클릭하여 블럭에 할당을 추가합니다. 마우스 오른쪽 버튼을 클릭하고 상황에 맞는 메뉴에서 **할당 추가(Add assignment)**를 선택합니다. "???" 평선블럭의 출력을 클릭하고 변수 이름으로 "GREEN"을 지정합니다.

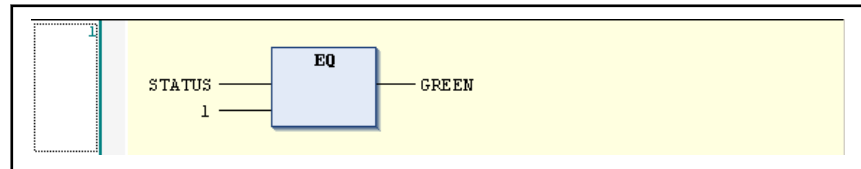


그림 4-24: "EQ" 평선블럭

변수 선언 창이 열립니다.

- "GREEN"은 평선블럭의 출력 변수이므로 제안을 "VAR\_OUTPUT, GREEN, BOOL"로 변경해야 합니다.

평선블럭의 선언영역에 있는 "VAR\_OUTPUT...END\_VAR" 영역에 선언 줄 GREEN: BOOL;이 나타납니다.

**"EQ"의 기능**

"STATUS" 입력 변수를 "1"과 비교합니다. "STATUS"의 값에 "1"이 있는 경우 결과는 "TRUE"입니다. "STATUS"의 값에 "1"이 없는 경우 결과는 "FALSE"입니다. 결과는 출력 변수 "GREEN"에 할당됩니다.

**네트워크 추가**

"노란색" 및 "빨간색" 신호 단계에 대해서도 "STATUS" 입력 변수의 평가를 구현해야 합니다. 따라서 두 개의 네트워크를 더 추가해야 합니다.

- 첫 번째 네트워크를 클릭하고 상황에 맞는 메뉴에서 **네트워크 추가(아래)(Add network (below))**를 선택합니다.

**두 번째 네트워크 편집**

두 번째 네트워크에서 "EQ" 평선블럭을 만듭니다.

- 두 번째 네트워크에서 마우스 오른쪽 버튼을 클릭합니다(오른쪽 클릭).

첫 단계 - IndraLogic

2. 상황에 맞는 메뉴에서 **FB(평선블럭) 호출 삽입(Insert FB call)**을 선택합니다.
3. "FB(평선블럭) 호출 삽입"(Insert FB call)에 대한 "입력 보조툴"(Input Assistance)의 "범주:"(Categories:) 아래에서 "키워드"(Keywords)를 선택합니다.  
"요소:"(Elements:) 아래에서 "EQ" 요소를 선택합니다.  
"확인"(OK) 버튼을 클릭하여 확인합니다.
4. "EQ" 평선블럭의 출력을 강조하고 마우스 오른쪽 버튼을 클릭합니다.
5. 상황에 맞는 메뉴에서 **FB(평선블럭) 호출 삽입(Insert FB call)**을 선택합니다.
6. "FB(평선블럭) 호출 삽입"(Insert FB call)에 대한 "입력 보조툴"(Input Assistance)의 "범주:"(Categories:) 아래에서 "키워드"(Keywords)를 선택합니다.  
"요소:"(Elements:) 아래에서 "OR" 요소를 선택합니다.  
"확인"(OK) 버튼을 클릭하여 확인합니다.

"OR" 평선블럭의 추가 할당을 위해 다음 단계를 진행합니다.

1. "OR" 평선블럭의 출력을 강조하고 마우스 오른쪽 버튼을 클릭합니다.
2. 출력에서 상황에 맞는 메뉴 **할당 삽입(Insert assignment)**을 통해 삽입합니다.
3. "???" 평선블럭의 출력에서 출력 변수 이름으로 "YELLOW"를 입력합니다.
4. 창에서 입력을 완료하여 "YELLOW" 변수를 선언합니다.  
평선블럭의 선언영역에 있는 "VAR\_OUTPUT...END\_VAR" 영역에 선언 줄 YELLOW: BOOL; 이 나타납니다.

1. "???" 오른쪽에 있는 연결에서 평선블럭을 마우스 오른쪽 버튼으로 클릭하여 "OR" 평선블럭의 두 번째 (자유) 입력을 선택합니다.
2. 상황에 맞는 메뉴에서 **FB(평선블럭) 호출 삽입(Insert FB call)**을 선택합니다.
3. "FB(평선블럭) 호출 삽입"(Insert FB call)에 대한 "입력 보조툴"(Input Assistance)의 "범주:"(Categories:) 아래에서 "키워드"(Keywords)를 선택합니다.  
"요소:"(Elements:) 아래에서 "EQ" 요소를 선택합니다.  
"확인"(OK) 버튼을 클릭하여 확인합니다.

다음과 같이 두 개의 "EQ" 평선블럭에 대한 입력을 계속해서 할당해야 합니다.

1. 위쪽 "EQ" 평선블럭의 첫 번째 입력에서 두 번째 네트워크를 클릭합니다.
2. "STATUS"를 입력하고 <Enter> 키를 눌러 확인합니다.



"STATUS"는 이미 선언되었으므로 이번에는 선언 창이 나타나지 않습니다.

3. 동일한 "EQ" 평선블럭의 두 번째 입력을 클릭합니다.  
"2"를 입력하고 <Enter> 키를 눌러 확인합니다.
4. 아래쪽 "EQ" 평선블럭의 첫 번째 입력에서 두 번째 네트워크를 클릭합니다.  
"STATUS"를 입력하고 <Enter> 키를 눌러 확인합니다.
5. 동일한 "EQ" 평선블럭의 두 번째 입력을 클릭합니다.  
"4"를 입력하고 <Enter> 키를 눌러 확인합니다.

## 첫 단계 - IndraLogic

## 네트워크 복사

세 번째 네트워크는 두 번째 네트워크를 복사하면 되므로 가장 쉽게 만들 수 있습니다.

네트워크를 복사하려면 다음 단계를 진행합니다.

1. 두 번째 네트워크를 강조하고 마우스 오른쪽 버튼을 클릭하여 상황에 맞는 메뉴를 엽니다.
2. 상황에 맞는 메뉴에서 **복사(Copy)**를 선택합니다.
3. 마우스 오른쪽 버튼을 클릭하여 상황에 맞는 메뉴를 다시 엽니다.
4. **추가(Add)**를 선택합니다.

복사된 네트워크가 네트워크 "2" 아래에 숫자 "3"으로 추가됩니다.

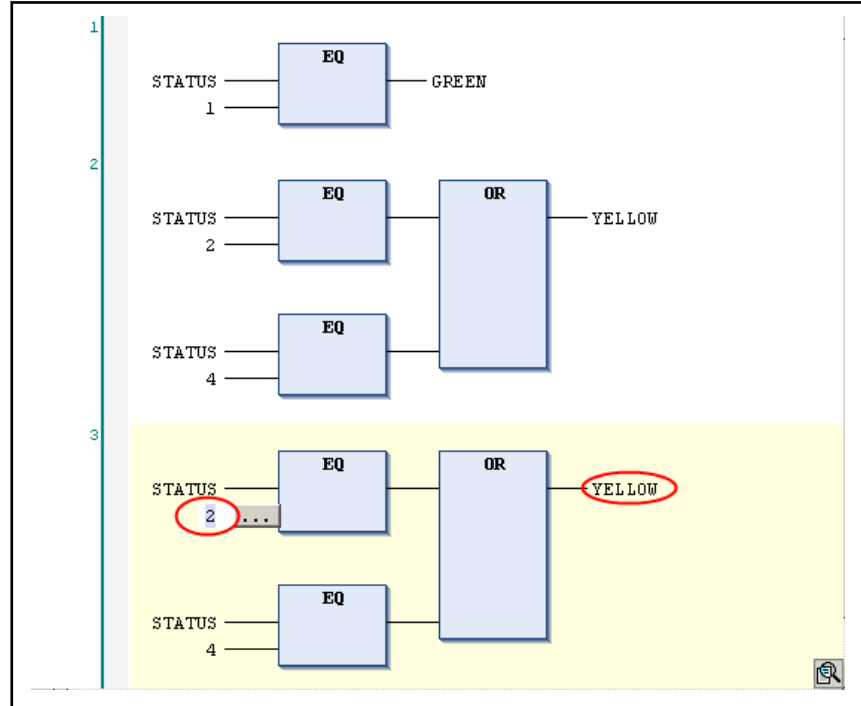


그림 4-25: 복사하여 추가된 세 번째 네트워크

## 세 번째 네트워크 편집

세 번째 네트워크를 편집하려면 다음 단계를 진행합니다.

1. 네트워크 3의 위쪽 "EQ 평선블록"에서 아래쪽 입력 "2"를 선택합니다.
2. 마우스 왼쪽 버튼으로 "2"를 클릭하고 "3"을 입력합니다.
3. "OR" 블록의 출력에서 마우스 왼쪽 버튼으로 변수 할당 "YELLOW"를 클릭하고 "RED"를 입력합니다.
4. 창에서 입력을 완료하여 "RED" 변수를 선언합니다.  
평선블록의 선언영역에 있는 "VAR\_OUTPUT...END\_VAR" 영역에 선언 줄 RED: BOOL; 이 나타납니다.

첫 번째 POU(프로그램 유닛) 평선블록 "TRAFFICSIGNAL"이 완료되었습니다.

첫 단계 - IndraLogic

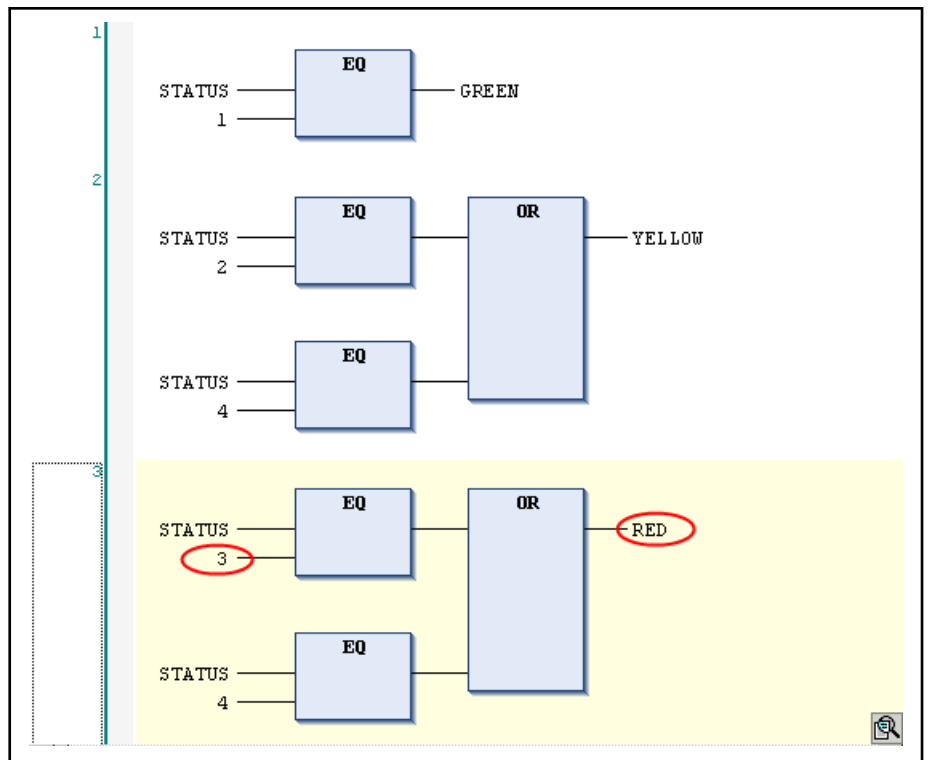


그림 4-26: 완료된 "TRAFFICSIGNAL" POU(프로그램 유닛) 구현을 위해 선언영역이 부수적으로 생성됩니다.

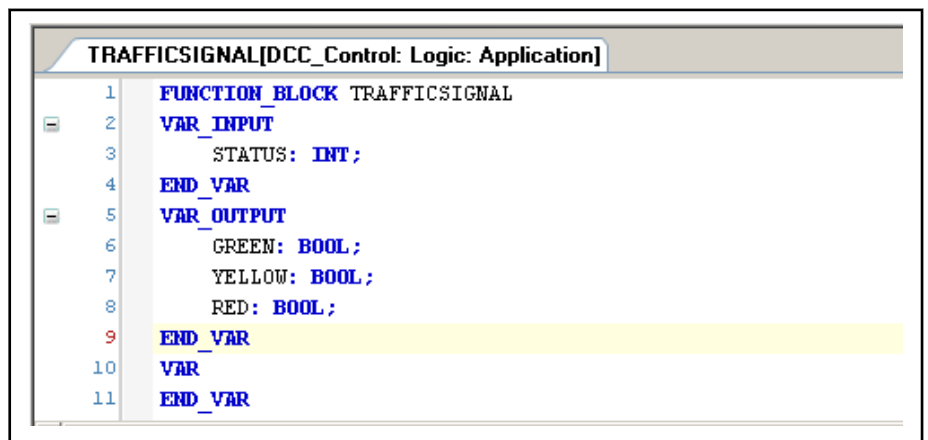


그림 4-27: "TRAFFICSIGNAL" 평선블럭의 선언영역

## "ST(구조체언어)"에서 "WAIT" 평선블럭 편집

### POU(프로그램 유닛) 열기

프로젝트 탐색기에서 개체를 두 번 클릭하여 POU(프로그램 유닛) "WAIT"를 엽니다.

### 변수 선언 및 사전 할당

"WAIT" 평선블럭은 입력 변수로 "TIME" 유형의 "WAITINGTIME" 변수를 받습니다. "WAIT" 평선블럭은 출력으로 "BOOL" 데이터 유형의 "OK" 변수를 제공합니다.

평선블럭 "TP" ZAB: TP;의 인스턴스가 필요합니다.

평선블럭의 현재 시간은 ETVAR: TIME; 변수에 기록됩니다.

## 첫 단계 - IndraLogic

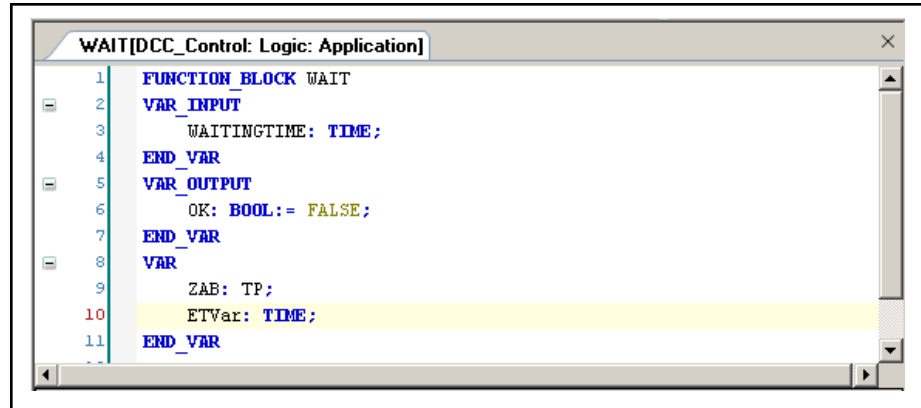


그림 4-28: "WAIT" POU(프로그램 유닛) 변수 선언

## 표준 라이브러리의 "TP" 평선블럭

펄스 발생기 기능이 있는 "TP" 평선블럭은 IndraWorks 표준 라이브러리에 있습니다. 라이브러리 관리자에서 "표준"(Standard) 라이브러리의 콘텐츠에 액세스하려면 다음 단계를 진행합니다.

1. 프로젝트 탐색기를 두 번 클릭하여 "라이브러리 관리자"(Library Manager)를 엽니다.
2. "라이브러리 관리자"(Library Manager)에서 "표준"(Standard) 라이브러리를 강조합니다.
3. "타이머"(Timer) 폴더 왼쪽의 "+"를 클릭하여 "타이머"(Timer) 폴더의 트리 구조를 엽니다.

트리에 "TP" 평선블럭이 있습니다. "TP" 평선블럭을 클릭하면 자세한 정보가 표시됩니다.

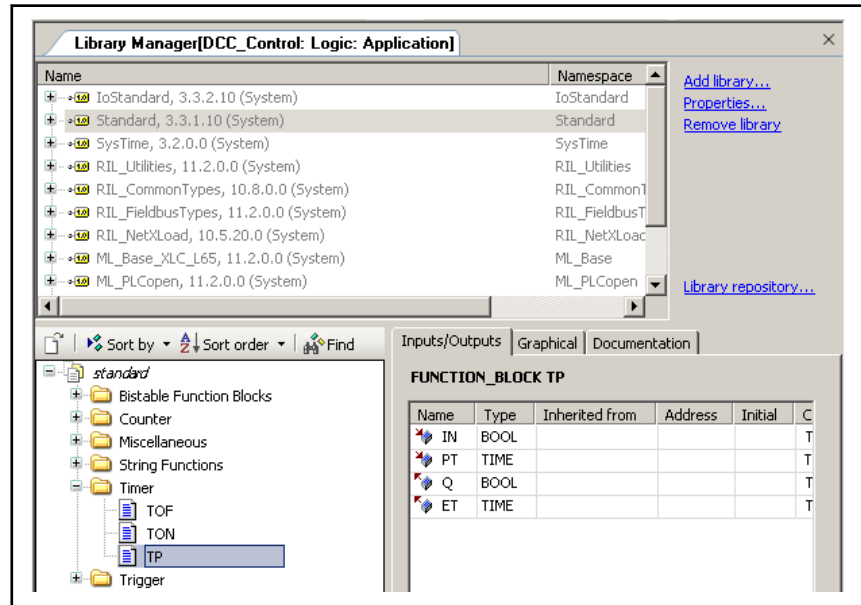


그림 4-29: 라이브러리 관리자 및 표준 평선블럭 "TP"

"TP" 평선블럭에는 두 개의 입력("IN" 및 "PT")과 두 개의 출력("Q" 및 "ET")이 있습니다.

**"TP"의 기능**

"IN"에 부울 값 "FALSE"가 있는 한 "ET" = "0" 및 "Q"는 "FALSE"입니다.

"IN"이 부울 값 "TRUE"를 제공하는 즉시 "ET" 출력에 시간(밀리초)이 추가됩니다.

"ET"가 "PT"의 값에 도달하면 "ET"는 계산을 중지합니다.

"ET"가 "PT"보다 작은 한 "Q"는 "TRUE"를 제공합니다.

## 첫 단계 - IndraLogic

변수 "PT"의 값에 도달하는 즉시 "Q"는 부울 값 "FALSE"를 제공합니다.



다음과 같이 변수로 복사하여 평선블럭 인스턴스의 출력을 평가할 수 있습니다.

ZAB(IN:=TRUE,PT:=WAITINGTIME,ET=>ETVar);

또는 다음과 같이 점 연산자를 사용할 수도 있습니다.

**ZAB.Q**

ST(구조체언어)에서 구현 만들기

다음에 표시된 대로 구현 섹션을 완료합니다.

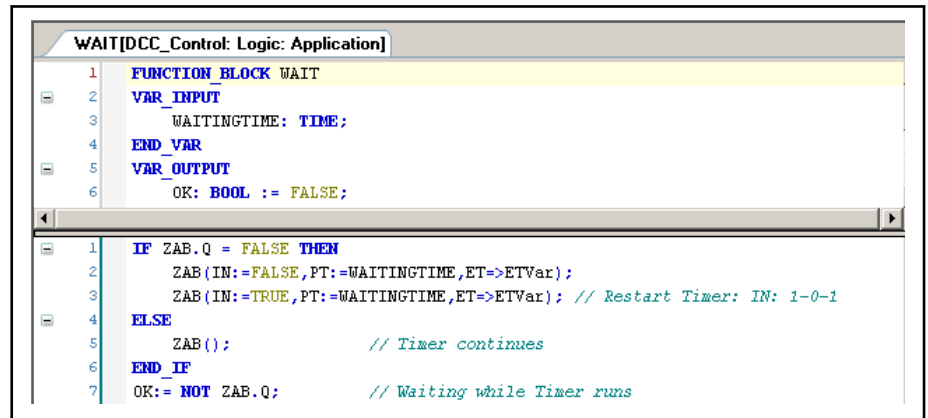


그림 4-30: ST(구조체언어)에서 완료된 "WAIT" POU(프로그램 유닛)

"WAIT"의 기능

시간 단계가 완료된 경우 "IN"의 1-0-1 신호 변경 및 현재 대기 시간 "WAITINGTIME"을 사용하여 다시 시작됩니다. 그렇지 않은 경우 시간 단계는 출력을 업데이트하여 계속됩니다.

시간 단계가 완료된 후에(ZAB.Q = 0) PLC 주기에 대해 "OK" 출력의 값은 "TRUE"입니다.

## "AS"에서 "SEQUENCE" 프로그램 편집

프로그램 열기

프로젝트 탐색기에서 개체를 두 번 클릭하여 "SEQUENCE" POU(프로그램 유닛)를 엽니다.

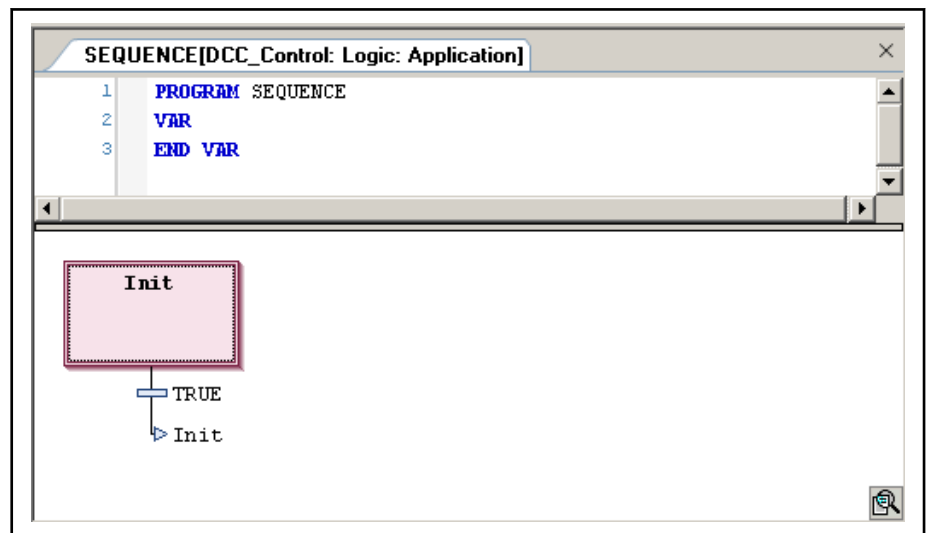


그림 4-31: "SEQUENCE" 프로그램의 편집기 열기

변수 선언

"SEQUENCE" 평선블럭은 입력 변수로 "BOOL" 유형의 "START" 변수를 받습니다.

## 첫 단계 - IndraLogic

이 평션블럭은 "INT" 유형의 "TRAFFICSIGNAL1" 및 "TRAFFICSIGNAL2"를 제공하고 로컬 변수로 "WAIT" 유형의 "DELAY" 변수를 제공합니다. 먼저 DELAY: WAIT; 변수만 지정합니다. 다른 변수는 나중에 구현 섹션에서 입력 보조틀을 사용하여 선언합니다.

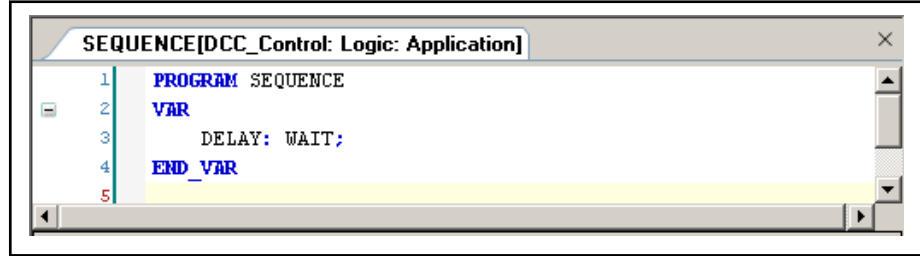


그림 4-32: "SEQUENCE" POU(프로그램 유닛)의 변수 선언 - 첫 번째 부분

### 사용자 프로그램 만들기

시퀀스 캐스케이드가 시작될 때 "Init" 작업, 이후의 "TRUE" 전환 및 "Init"로의 반환은 "SFC 편집기"의 본문에 있습니다. ["SEQUENCE" POU\(프로그램 유닛\)의 편집기 열기\(43 페이지\)](#)를 참조하십시오.

### 시퀀스 캐스케이드의 구조 지정

먼저 각 단계 및 전환을 추가하여 시퀀스 캐스케이드의 구조를 지정합니다. 필요한 단계와 전환을 추가하면 단계 및 전환에 대한 각 작업이 프로그래밍됩니다. 단계는 각 신호 단계에 필요합니다. 단계를 추가한 후 이어서 전환을 추가하려면 다음 단계를 진행합니다.

1. "TRUE" 전환을 강조하고 마우스 오른쪽 버튼을 클릭하여 상황에 맞는 메뉴를 엽니다.
2. 상황에 맞는 메뉴에서 **나중에 단계 전환 추가(Add step transition afterward)**를 선택합니다.
3. 마지막 전환을 각각 강조하고 단계와 전환을 세 개 더 추가합니다.

시퀀스 캐스케이드가 다음과 같이 표시됩니다.

첫 단계 - IndraLogic

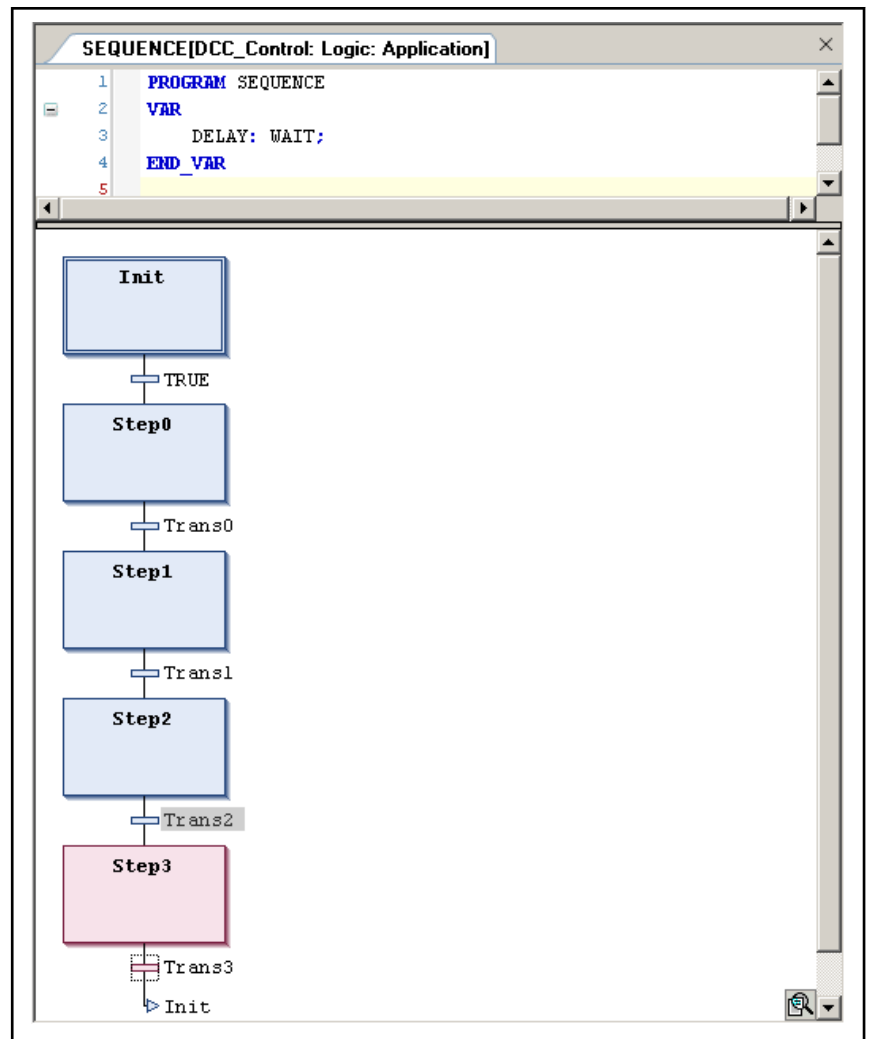


그림 4-33: 추가된 "SEQUENCE" 프로그램 단계



또는 SFC ▶ Add step transition afterward(나중에 단계 전환 추가)를 사용하여 단계와 전환을 추가할 수 있습니다.

전환 프로그래밍

전환을 프로그래밍하는 방법에는 다음 두 가지가 있습니다.

1. 전환 이름 변경: 진행 조건이 하나만 필요한 경우(예: 변수 조회)에만 유용합니다.
2. 전환 프로그래밍: 연결된 변수의 결과가 진행 조건으로 의도된 것일 경우 적절합니다.

전환을 프로그래밍하려면 전환을 추가해야 합니다. 여기에서는 "전환"을 두 번 클릭하고 이름을 지정한 다음 전환을 작성할 실행언어(프로그래밍 언어)를 지정하여 전환을 추가합니다. 그런 다음 기능과 마찬가지로 편집기에서 전환을 프로그래밍할 수 있습니다.

첫 번째 전환으로 "START" 전환을 선언합니다. "START" 전환을 선언하려면 다음 단계를 진행합니다.

1. 첫 번째 전환에 해당하는 전환 이름 "TRUE"를 클릭합니다.
2. 입력 필드에 "START"를 입력합니다.

"START" 변수는 아직 선언영역에서 선언되지 않았기 때문에 변수 선언에 필요한 창이 열립니다.

## 첫 단계 - IndraLogic

The 'Auto Declare' dialog box is shown with the following fields and values:

- Scope: VAR
- Name: START
- Type: BOOL
- Object: SEQUENCE [DCC\_Control: Log]
- Initialization: (empty)
- Address: (empty)
- Flags:
  - ☐ CONSTANT
  - ☐ RETAIN
  - ☐ PERSISTENT
- Comment: (empty text area)

Buttons: OK, Cancel

그림 4-34: 변수 선언에 필요한 입력 보조틀

3. "유형:"(Type:) 아래에서 "BOOL" 데이터 유형을 선택합니다.
4. "표시 유형:"(Visibility:)에서 "VAR\_INPUT" 변수 유형을 선택합니다.
5. "확인"(OK)을 클릭하여 확인합니다.

"START" 변수가 첫 번째 전환에 할당됩니다. "START" = "TRUE"이면 다음 작업이 수행됩니다. 진행 조건이 충족됩니다.

두 번째 및 이후의 모든 전환에 대해 다음 단계를 진행합니다.

1. 전환 이름을 클릭합니다.
2. 전환 이름을 DELAY.OK 로 변경합니다.

"DELAY" 또는 "WAIT" 평선블럭의 "OK" 출력 변수 값이 "TRUE"이면 다른 전환이 이행됩니다.

시퀀스 캐스케이드가 다음과 같이 표시됩니다.

첫 단계 - IndraLogic

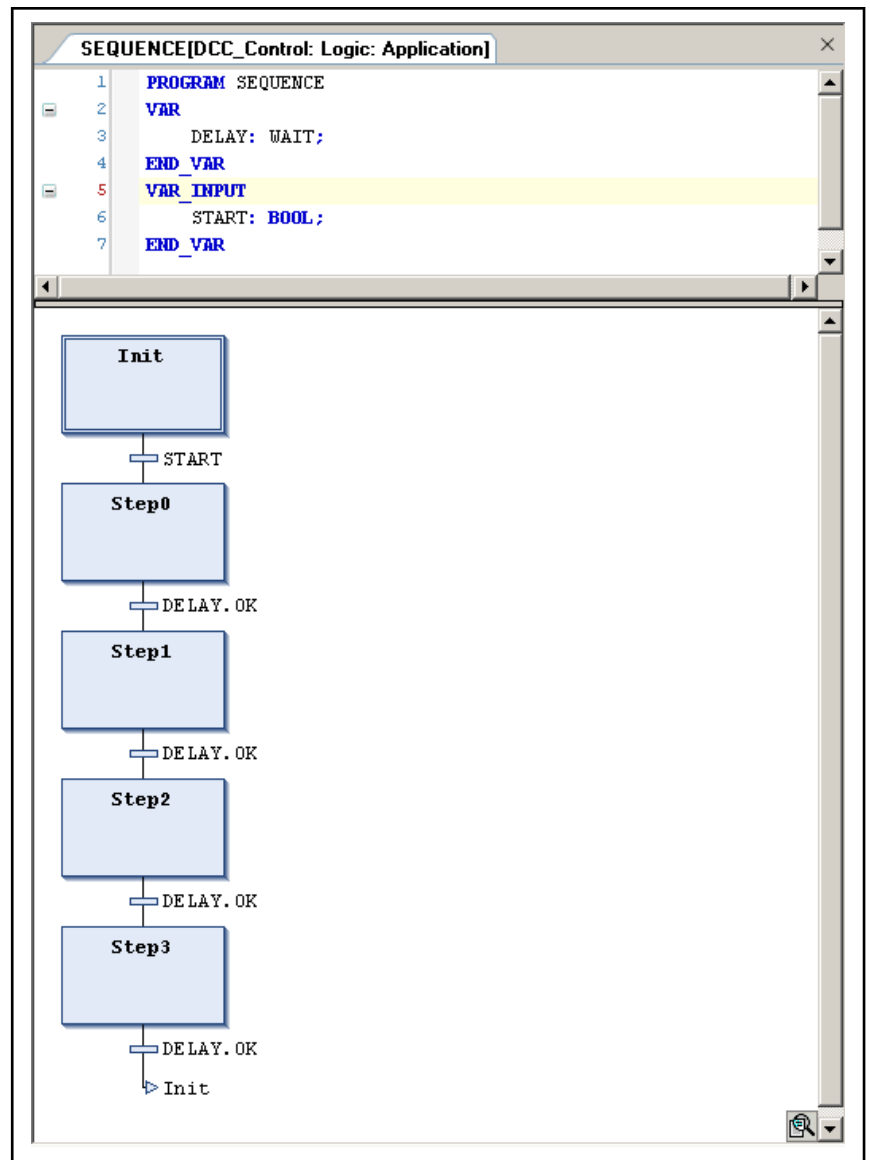


그림 4-35: 시퀀스 캐스케이드의 단계

- 단계 이름 지정** "Init" 단계만 원래 이름을 유지합니다. 나머지 단계는 이름이 바뀝니다. 연속해서 "Switch1", "Green2", "Switch2" 및 "Green1"로 이름을 지정해야 합니다. 각 단계의 이름을 바꾸려면 다음 단계를 진행합니다.
1. 단계의 이름을 클릭합니다.
  2. 원하는 이름을 입력합니다.
  3. "Enter" 키를 눌러 확인합니다.
- 단계의 기능** "Switch1" 및 "Switch2" 단계에서는 "노란색 단계"가 구현됩니다. "Green1" 단계에서는 교통 신호 1에 대한 녹색 단계가 구현되고 "Green2"에서는 교통 신호 2에 대한 녹색 단계가 구현됩니다.
- 반환 주소 변경** 단계가 실행된 후에는 더 이상 "Init" 다음에 반환이 없습니다. 대신 "Switch1" 단계로 이동해야 합니다. "Init" 반환의 이름을 클릭하고 여기에 새 반환 대상 "Switch1"을 입력합니다.
- 시퀀스 캐스케이드가 다음과 같이 표시됩니다.

## 첫 단계 - IndraLogic

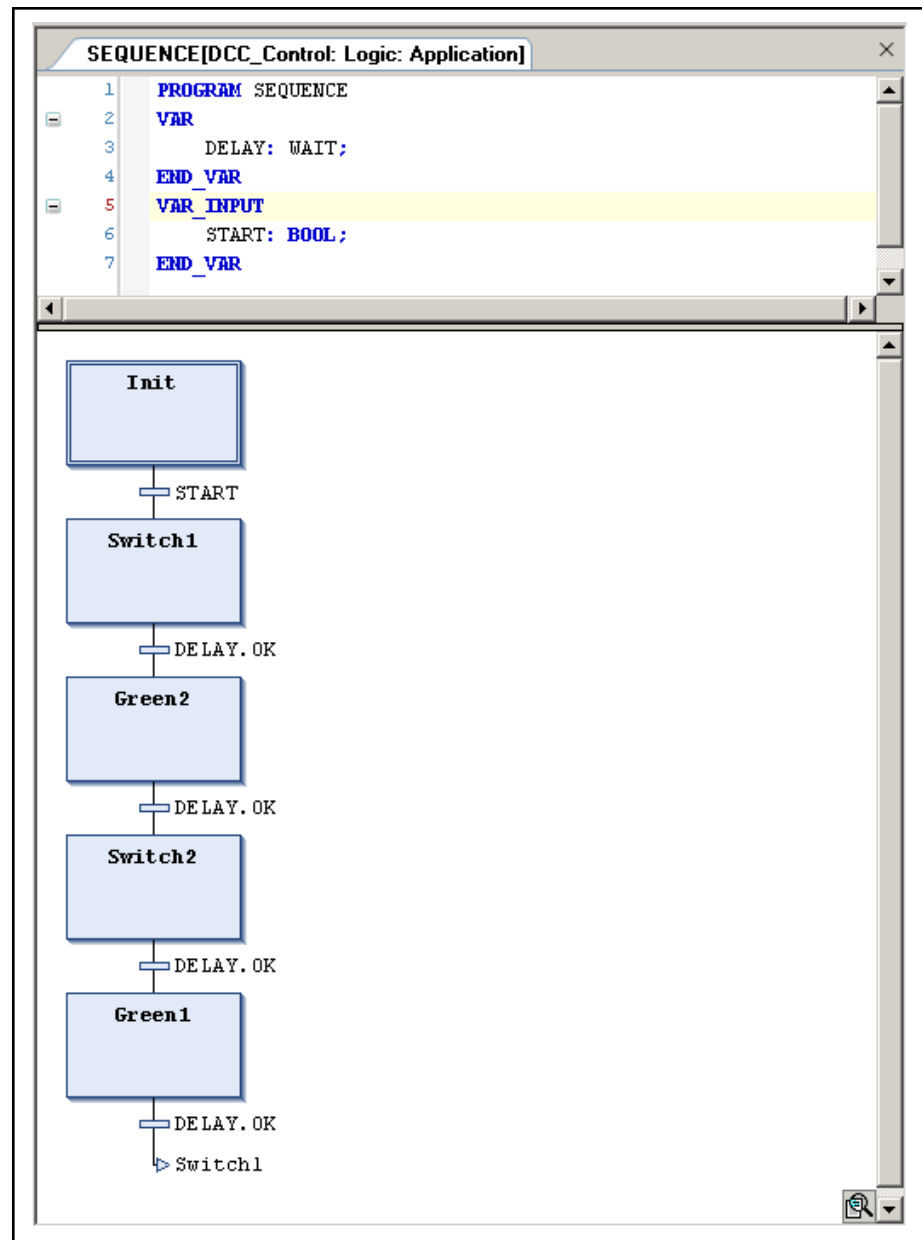


그림 4-36: 이름 지정 후 시퀀스 캐스케이드의 단계

## 단계 및 작업

SFC(시퀀셜 펄스 차트)에서 작성된 펄스블럭은 유도된 연결(전환)을 통해 서로 연결된 일련의 단계로 구성되어 있습니다. 단계에는 처리할 프로그램 코드가 저장된 작업이 할당됩니다.

"IndraLogic"은 다음과 같은 유형의 작업을 지원합니다.

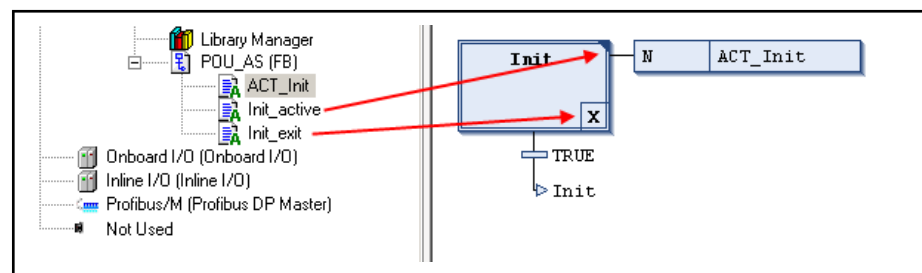


그림 4-37: 단계 작업의 유형

## 첫 단계 - IndraLogic

- **입력 작업:** 입력 작업은 단계가 활성화될 때 정확히 한 번 처리됩니다(이 경우에는 "Init\_active"). 입력 작업이 할당된 단계의 오른쪽 위에 검정색 삼각형이 표시됩니다.
- **연결된 작업:** 작업은 POU(프로그램 유닛)의 구성 요소로 개별적으로 생성되고 필요한 단계와 연관 지어 연결됩니다(이 경우에는 "ACT\_Init"). 각 단계의 시간 동작은 확인 기호(이 경우 N)에 의해 결정됩니다. 연결된 작업은 단계의 오른쪽에 표시되고, 이러한 작업이 비활성화되면 정확히 한 번 사후 처리됩니다.
- **출력 작업:** 출력 작업은 단계가 비활성화되기 전에 한 번만 실행됩니다(이 경우에는 "Init\_exit"). 출력 작업이 할당된 단계의 오른쪽 아래에는 "X"가 표시됩니다.

### "ST(구조체언어)"로 "Init" 작업 프로그래밍

"Init" 단계에 대한 작업에서는 "TRAFFICSIGNAL1" 및 "TRAFFICSIGNAL2" 변수가 초기화됩니다. 교통 신호 1에 대한 "TRAFFICSIGNAL1"의 상태는 "1"(녹색)이어야 합니다. 교통 신호 2에 대한 "TRAFFICSIGNAL2"의 상태는 "3"(빨간색)이어야 합니다.

"Init"를 편집하려면 다음 단계를 진행합니다.

1. "SEQUENCE" 프로그램에서 첫 번째 "Init" 단계의 상자를 두 번 클릭합니다.  
"작업 추가"(Add action) 창이 열립니다.
2. 여기에서 실행언어(프로그래밍 언어)로 "ST(구조체언어)"를 선택합니다
3. "열기"(Open) 버튼을 사용하여 설정을 확인합니다.

프로젝트 탐색기에서 "SEQUENCE (FB)" POU(프로그램 유닛) 아래에 "Init\_active" 작업 분기가 생성됩니다.

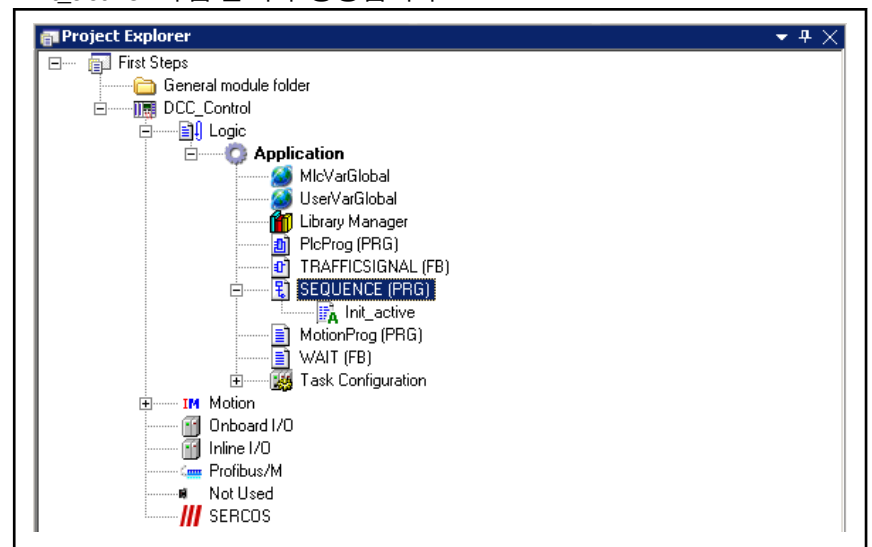


그림 4-38: 프로젝트 탐색기에서 "Init\_active" 작업이 있는 "SEQUENCE" POU(프로그램 유닛)  
"Init\_active" 작업의 편집기를 엽니다.

4. 편집기에 다음 프로그램 줄을 입력합니다.

"Init\_active" 작업

```
TRAFFICSIGNAL1 := 1;
TRAFFICSIGNAL2 := 3;
```

## 첫 단계 - IndraLogic



"Init\_active" 작업을 프로그래밍할 때 "TRAFFICSIGNAL1" 및 "TRAFFICSIGNAL2" 변수는 아직 알 수 없습니다. "변수 선언"(Declaring variables) 창이 자동으로 열립니다. 두 변수에 대해 모두 표시 유형으로 "VAR\_OUTPUT"를 선택하고 데이터 유형으로 "INT"를 선택합니다.

### "ST(구조체언어)"로 "Switch1" 작업 프로그래밍

"Init" 작업의 편집기는 다음과 같이 표시됩니다.

"Switch1" 단계에 대한 작업에서 교통 신호 1(TRAFFICSIGNAL1)의 STATUS가 "2"(노란색)로 변경되고 교통 신호 2(TRAFFICSIGNAL2)의 STATUS는 "4"(노란색-빨간색)로 변경됩니다. 2 초의 지연 시간이 지정됩니다.

"Switch1"을 편집하려면 다음 단계를 진행합니다.

1. "SEQUENCE" POU(프로그램 유닛)에서 두 번째 단계 "Switch1"의 상자를 두 번 클릭합니다.

"작업 추가"(Add action) 창이 열립니다.

2. 여기에서 실행언어(프로그래밍 언어)로 "ST(구조체언어)"(Structured Text (ST))를 선택합니다.

3. "열기"(Open) 버튼을 사용하여 입력을 확인합니다.

프로젝트 탐색기에서 "SEQUENCE (FB)" POU(프로그램 유닛) 아래에 "Switch1\_active" 작업 분기가 생성됩니다.

"Switch1\_active" 작업의 편집기가 열립니다.

4. 편집기에 다음 프로그램 줄을 입력합니다.

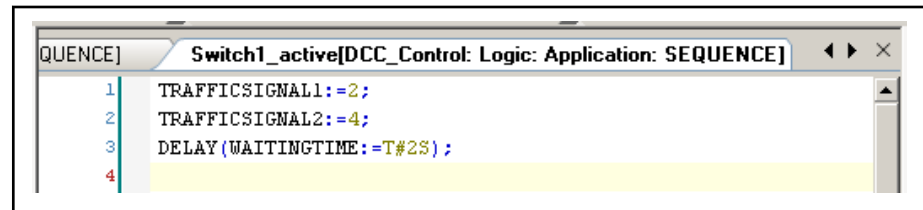


그림 4-39: "Switch1\_active" 작업의 편집기

### "ST(구조체언어)"로 "Green2" 작업 프로그래밍

"Green2" 단계에 대한 작업에서 교통 신호 1(TRAFFICSIGNAL1)의 STATUS가 "3"(빨간색)으로 변경되고 교통 신호 2(TRAFFICSIGNAL2)의 STATUS는 "1"(녹색)로 변경됩니다. 5 초의 지연 시간이 지정됩니다.

"Green2"를 편집하려면 다음 단계를 진행합니다.

1. "SEQUENCE" POU(프로그램 유닛)에서 세 번째 단계 "Green2"의 상자를 두 번 클릭합니다.

"작업 추가"(Add action) 창이 열립니다.

2. 여기에서 실행언어(프로그래밍 언어)로 "ST(구조체언어)"(Structured Text (ST))를 선택합니다.

3. "열기"(Open) 버튼을 사용하여 입력을 확인합니다.

프로젝트 탐색기에서 "SEQUENCE (FB)" POU(프로그램 유닛) 아래에 "Green2\_active" 작업 분기가 생성됩니다.

"Green2\_active" 작업의 편집기를 엽니다.

4. 편집기에 다음 프로그램 줄을 입력합니다.

첫 단계 - IndraLogic

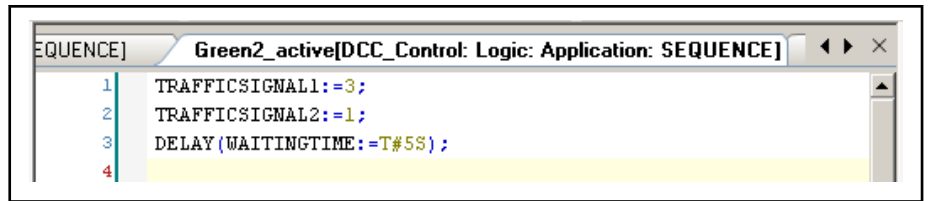


그림 4-40: "Green2\_active" 작업의 편집기

"ST(구조체언어)"로 "Switch12" 작업 프로그래밍

"Switch2" 단계에 대한 작업에서 교통 신호 1(TRAFFICSIGNAL1)의 STATUS 가 "4"(노란색-빨간색)로 변경되고 교통 신호 2(TRAFFICSIGNAL2)의 STATUS 는 "2"(노란색)로 변경됩니다. 2 초의 지연 시간이 지정됩니다.

"Switch2"를 편집하려면 다음 단계를 진행합니다.

1. "SEQUENCE" POU(프로그램 유닛)에서 네 번째 단계 "Switch2"의 상자를 두 번 클릭합니다.  
"작업 추가"(Add action) 창이 열립니다.
2. 여기에서 실행언어(프로그래밍 언어)로 "ST(구조체언어)"(Structured Text (ST))를 선택합니다.
3. "열기"(Open) 버튼을 사용하여 입력을 확인합니다.  
프로젝트 탐색기에서 "SEQUENCE (FB)" POU(프로그램 유닛) 아래에 "Switch2\_active" 작업 분기가 생성됩니다.  
"Switch2\_active" 작업의 편집기가 열립니다.
4. 편집기에 다음 프로그램 줄을 입력합니다.

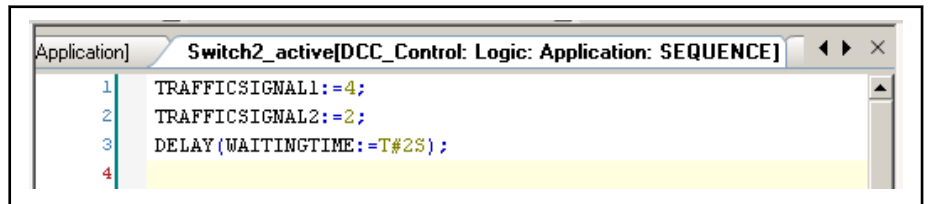


그림 4-41: "Switch2\_active" 작업의 편집기

"ST(구조체언어)"로 "Green1" 작업 프로그래밍

"Green1" 단계에 대한 작업에서 TRAFFICSIGNAL 1 의 STATUS 가 "1"(녹색)로 변경되고 TRAFFIC SIGNAL 2 의 STATUS 는 "3"(빨간색)으로 변경됩니다. 5 초의 지연 시간이 지정됩니다.

"Green1"을 편집하려면 다음 단계를 진행합니다.

1. 다섯 번째 단계 "Green1"의 상자를 두 번 클릭합니다.  
"작업 추가"(Add action) 창이 열립니다.
2. 여기에서 실행언어(프로그래밍 언어)로 "ST(구조체언어)"(Structured Text (ST))를 선택합니다.
3. "열기"(Open) 버튼을 사용하여 입력을 확인합니다.  
프로젝트 탐색기에서 "SEQUENCE (FB)" POU(프로그램 유닛) 아래에 "Green1\_active" 작업 분기가 생성됩니다.  
"Green1\_active" 작업의 편집기를 엽니다.
4. 편집기에 다음 프로그램 줄을 입력합니다.

## 첫 단계 - IndraLogic

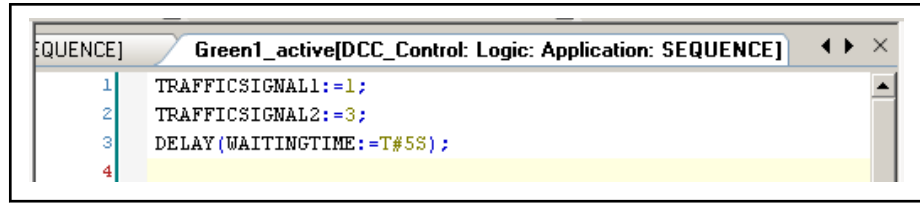


그림 4-42: "Green1\_active" 작업의 편집기

완료된 "SEQUENCE" 평선블럭

이제 프로젝트 탐색기의 "SEQUENCE" 평선블럭 및 구조가 완성되었습니다. 다음 그림을 참조하십시오.

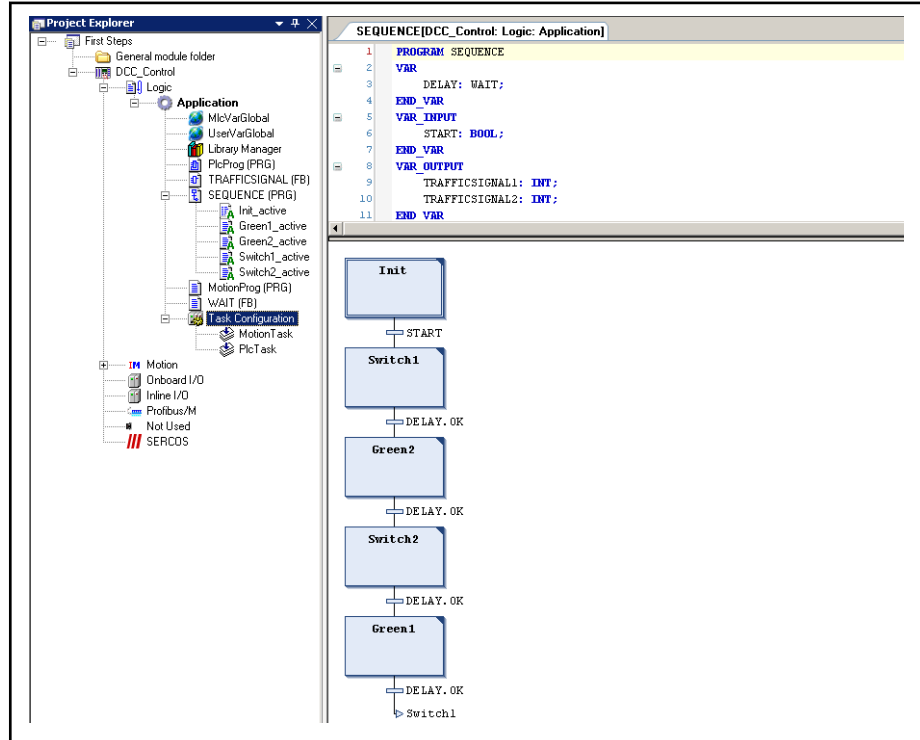


그림 4-43: 완료된 "SEQUENCE" POU(프로그램 유닛)

## 4.6.4 작업 구성

**작업** "작업"은 수행할 작업을 말합니다. 여기에서는 "순환" 또는 "자유"와 같은 "작업 유형"을 선언하고 런타임을 할당하여 프로그램을 호출합니다. 프로그램 시퀀스는 "작업"을 사용하여 구성됩니다.

**작업 구성자** 작업 구성자는 응용 프로그램의 실행을 제어하는 하나 이상의 작업을 정의합니다.

**"SEQUENCE" 작업 추가** 응용 프로그램에는 두 가지 "프로그램" 유형의 POU(프로그램 유닛), 즉 "PlcProg" 및 "SEQUENCE"가 있습니다. "PlcProg"는 컨트롤을 추가할 때 자동으로 생성됩니다. "PlcProg" 작업은 또한 "PlcProg" POU(프로그램 유닛)의 시퀀스를 지정할 수 있도록 컨트롤을 추가할 때 생성됩니다. "SEQUENCE" POU(프로그램 유닛)를 호출할 수 있도록 "SEQUENCE"에 대한 다른 작업을 추가합니다.

"SEQUENCE" 작업을 추가하려면 다음 단계를 진행합니다.

1. 프로젝트 탐색기에서 "작업 구성"(Task configuration) 아래를 두 번 클릭하여 "PlcTask" 구성을 엽니다.
2. "POUs"(프로그램 유닛) 섹션에서 열려 있는 구성 "POU(프로그램 유닛) 삽입"(Insert POU)을 선택합니다.

## 첫 단계 - IndraLogic

"입력 보조툴"(Input Assistance) 창이 자동으로 열립니다.

3. "입력 보조툴"(Input Assistance)의 "요소:(Elements:)" 아래에서 "SEQUENCE" 및 "PlcProg"가 표시될 때까지 "DCC\_Control" 아래의 트리를 엽니다.

4. "SEQUENCE"를 선택하고 "확인"(OK) 버튼을 클릭하여 확인합니다.

"Plc 작업 구성"의 모습은 이제 다음과 같습니다.

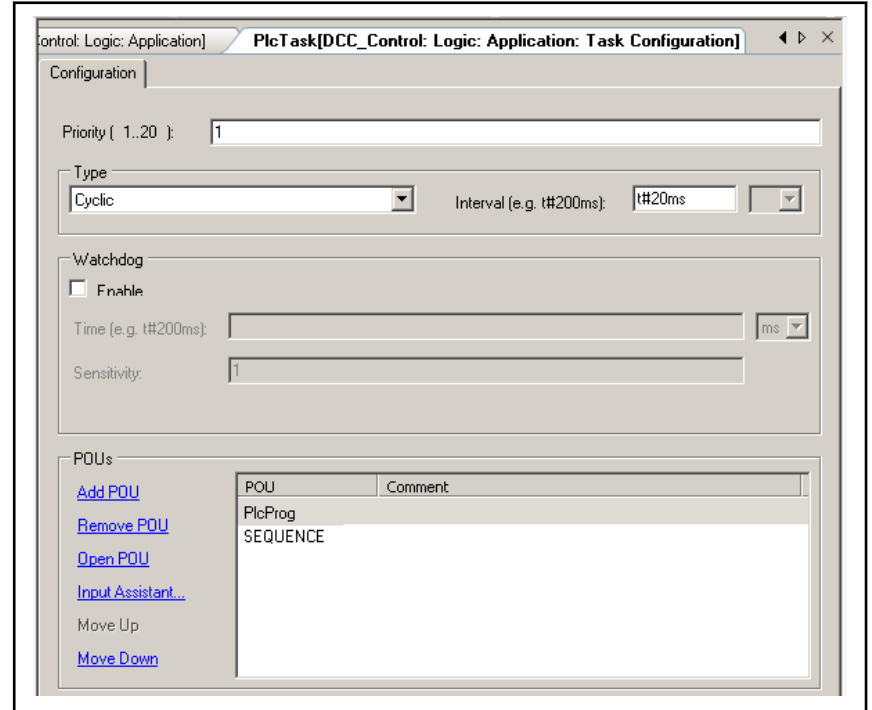


그림 4-44: 작업 구성

### 4.6.5 전역 변수 선언

GVL(전역변수리스트) 기능

신호등 시스템은 "ON" 스위치를 통해 전원이 켜집니다.

두 개의 신호등의 각 신호 해당하는 신호 단계의 프로그램 상태에 따라 제어됩니다.

두 신호등의 입력 하나와 출력 여섯 개에 대해 입력 및 출력 변수를 만들어야 합니다.

"%IX" 구문(입력의 경우)과 "%QX" 구문(출력의 경우)을 사용하여 입력 및 출력의 물리적 주소에 액세스할 수 있습니다.

크기 접두사 "X"는 주소에 액세스할 수 있는 영역을 설명합니다. 이 예제에서는 비트 주소에만 액세스할 수 있습니다.

변수는 "PlcProg" 평션블럭에서 뿐만 아니라 다른 평션블럭에서도 로컬로 사용할 수 있습니다. 따라서 변수는 "PlcProg" POU(프로그램 유닛)에서 선언하지 않으며 대신 전역변수리스트 "GVL"에서 선언합니다.

"GVL(전역변수리스트)" 만들기

"GVL(전역변수리스트)"을 프로젝트에 추가합니다("응용 프로그램"(Application)의 상황에 맞는 메뉴, **추가...(Add...)** ▶ **전역변수리스트...(Global Variable List...)**).



"응용 프로그램"(Application) 아래에 GVL(전역변수리스트)을 만들면 응용 프로그램의 모든 POU(프로그램 유닛)에서 변수를 사용할 수 있습니다.

## 첫 단계 - IndraLogic

주소를 사용하여 변수 선언 "전역변수리스트"(Global Variable List)에 새 "BOOL" 유형 변수가 표시됩니다. 다음 변수를 만듭니다.

*전역변수리스트 "GVL"*

```
VAR_GLOBAL
ON : BOOL;
A1_green AT %QX0.0 : BOOL;
A1_yellow AT %QX0.1 : BOOL;
A1_red AT %QX0.2 : BOOL;
A2_green AT %QX0.3 : BOOL;
A2_yellow AT %QX0.4 : BOOL;
A2_red AT %QX0.5 : BOOL;
END_VAR
```

변수 이름 다음에는 영역 접두사(출력의 경우 "Q", 입력의 경우 "I") 앞에 백분율 기호가 있는 "AT" 할당이 표시됩니다. 그 다음에는 비트 주소 지정을 위한 크기 접두사 "X"가 표시됩니다.

A1\_green AT %QX0.0: BOOL은 물리적 출력 비트 "Q 0.0"에 "BOOL" 유형의 "A1\_green" 변수를 할당합니다.



실제 컨트롤을 사용할 경우 이러한 출력이 정말로 존재하는지 확인하십시오.

XLC/MLC L45 및 L65 컨트롤에서 이러한 출력은 온보드 출력으로 표시됩니다(표준 매핑).

XLC/MLC L25 컨트롤은 인라인 모듈 또는 필드 버스를 통해 확장해야 합니다. 확장하지 않을 경우에는 절대 주소를 사용하지 마십시오. 그렇지 않으면 코드를 생성할 때 경고가 표시됩니다.

## 4.6.6 시뮬레이션 모드에서 "SEQUENCE" 테스트

"SEQUENCE" 프로그램 평선블럭이 이제 테스트할 수 있는 지점까지 완성되었습니다. 테스트는 시뮬레이션 모드에서 수행해야 합니다.

### "응용 프로그램" 컴파일

응용 프로그램을 컨트롤로 로드할 수 있으려면 먼저 "컴파일러"를 사용하여 실수 없이 응용 프로그램을 컴파일해야 합니다. 응용 프로그램을 컴파일하려면 **만들기(Create) ▶ 응용 프로그램 작성(Build Application)**을 선택합니다. 컴파일한 후에는 작업 창 아래의 "작업 목록"(Task list)에 "컴파일 완료 -- 오류 0 개, 경고 0 개: 다운로드 준비!"(Compiling completed -- 0 errors, 0 warnings: ready for download!)라는 메시지가 표시됩니다.

### 컨트롤 시뮬레이션

테스트할 수 있는 컨트롤이 없는 경우 컨트롤을 시뮬레이션할 수 있습니다. 컨트롤을 시뮬레이션하려면 "프로젝트 탐색기"(Project Explorer)에서 "DCC\_Control"을 클릭하고 마우스 오른쪽 버튼을 클릭하여 상황에 맞는 메뉴를 엽니다. **시뮬레이션(Simulation)**을 클릭하여 시뮬레이션 모드를 활성화합니다. 시뮬레이션이 활성화되면 "시뮬레이션"(Simulation)이 선택됩니다.

첫 단계 - IndraLogic

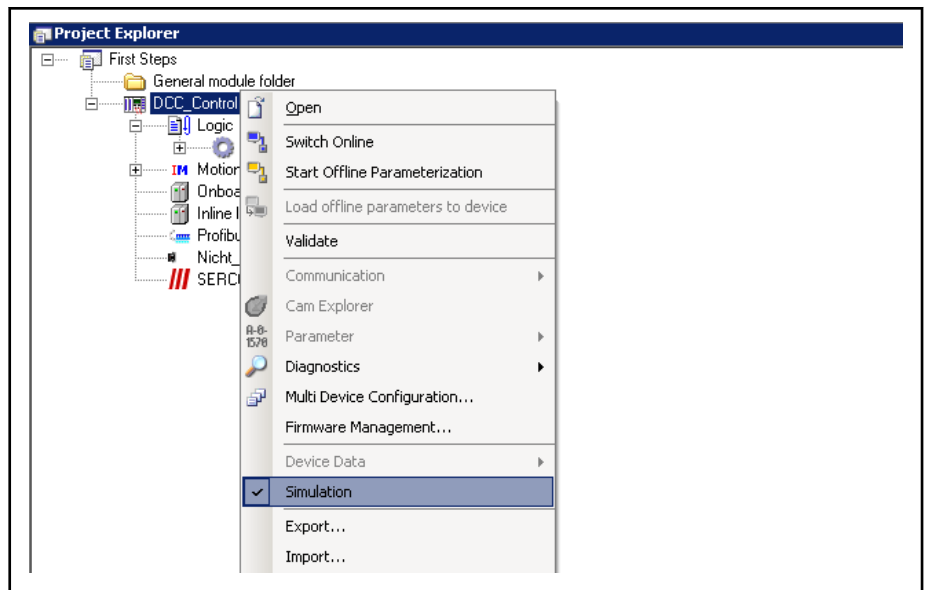


그림 4-45: 시뮬레이션 선택

컨트롤에 로그인

컨트롤에 로그인한 후 응용 프로그램을 컨트롤로 로드해야 합니다. 컨트롤에 로그인하려면 메뉴에서 디버그(Debug) ▶ 로그인(Login)을 선택합니다.

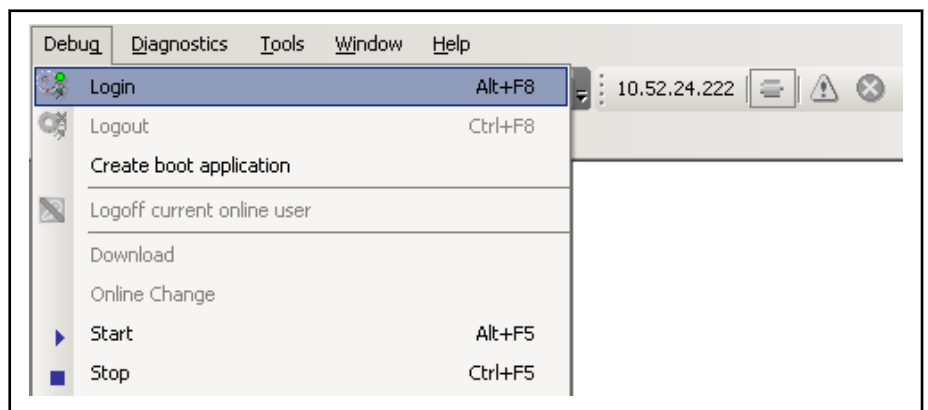


그림 4-46: 컨트롤에 로그인

프로젝트 탐색기가 다음과 같이 표시됩니다.

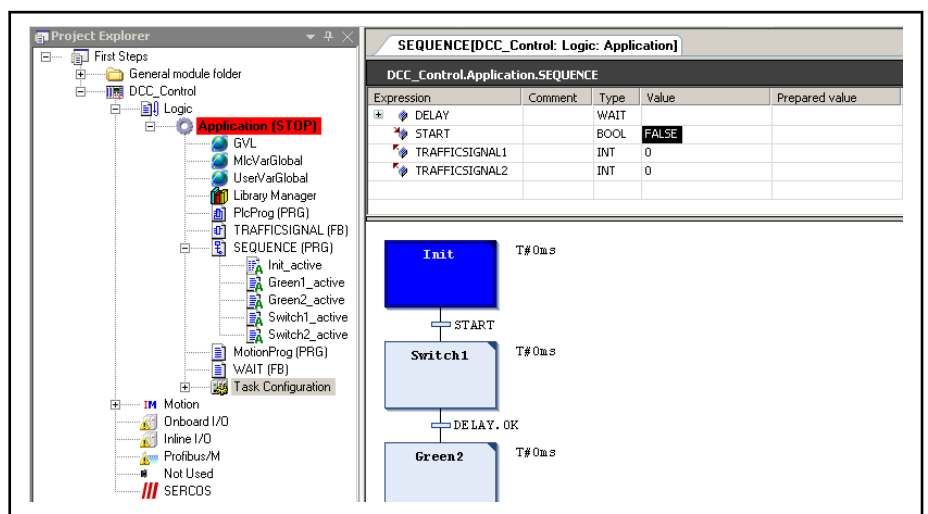


그림 4-47: 컨트롤에 로그인함(시뮬레이션) - 응용 프로그램 중지

## 첫 단계 - IndraLogic

**"응용 프로그램" 시작**

응용 프로그램을 시작하려면 응용 프로그램을 마우스 오른쪽 버튼으로 클릭하고 상황에 맞는 메뉴에서 **시작(Start)**을 선택하거나 주 메뉴에서 **디버그(Debug) ▶ 시작(Start)**을 선택합니다.

응용 프로그램이 컨트롤에서 실행됩니다.

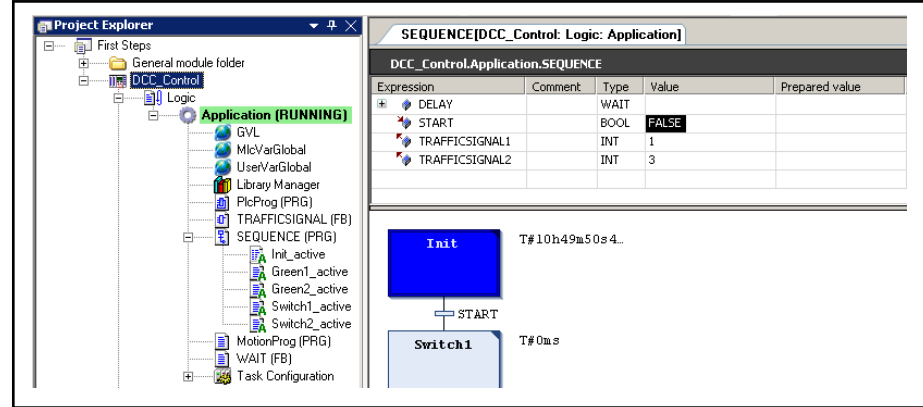


그림 4-48: 컨트롤에 로그인 - 응용 프로그램(실행 중)

**"SEQUENCE" 시작**

"SEQUENCE" POU(프로그램 유닛)를 엽니다. 선언의 온라인 모드에서 변수와 변수의 현재 값을 볼 수 있습니다. 선언에서 "TRUE"가 나타날 때까지 "준비된 값"(Prepared value) 열에서 "START" 변수 옆에 있는 줄을 클릭합니다.

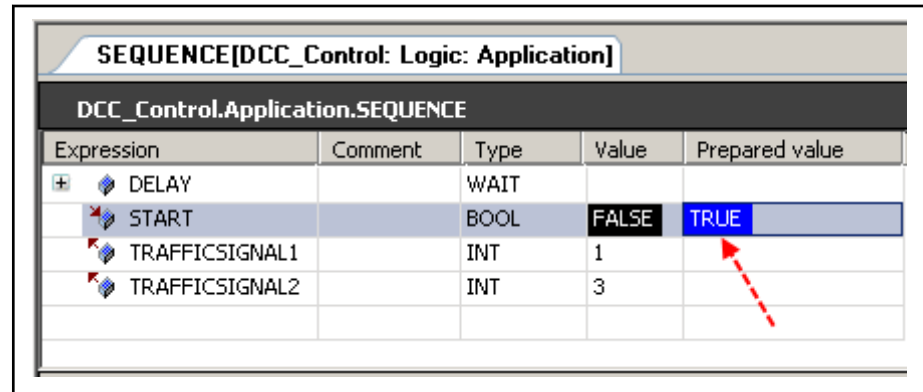


그림 4-49: "SEQUENCE" POU(프로그램 유닛) 선언 온라인

디버그(Debug) ▶ 값 쓰기(Write values)를 선택합니다(<Ctrl>+<F7>).

"START" 변수가 "TRUE"로 설정됩니다.

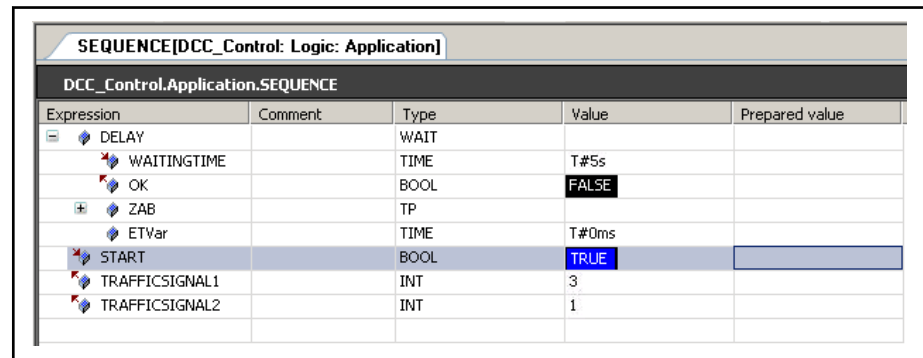


그림 4-50: "SEQUENCE" POU(프로그램 유닛) 선언 온라인

"SEQUENCE" POU(프로그램 유닛)의 개별 단계 및 전환이 처리됩니다. 마지막에 "Switch1" 단계로 이동합니다.

"START" 변수의 값은 나중에 "PlcProg" POU(프로그램 유닛)에서 설정됩니다.

## 첫 단계 - IndraLogic

### 컨트롤에서 로그아웃

컨트롤에서 로그아웃하기 전에 응용 프로그램을 중지합니다. 응용 프로그램을 중지하려면 응용 프로그램을 마우스 오른쪽 버튼으로 클릭하고 상황에 맞는 메뉴에서 **중지(Stop)**를 선택하거나 주 메뉴에서 **디버그(Debug) ▶ 중지(Stop)**를 선택합니다.

컨트롤에서 응용 프로그램이 중지됩니다.

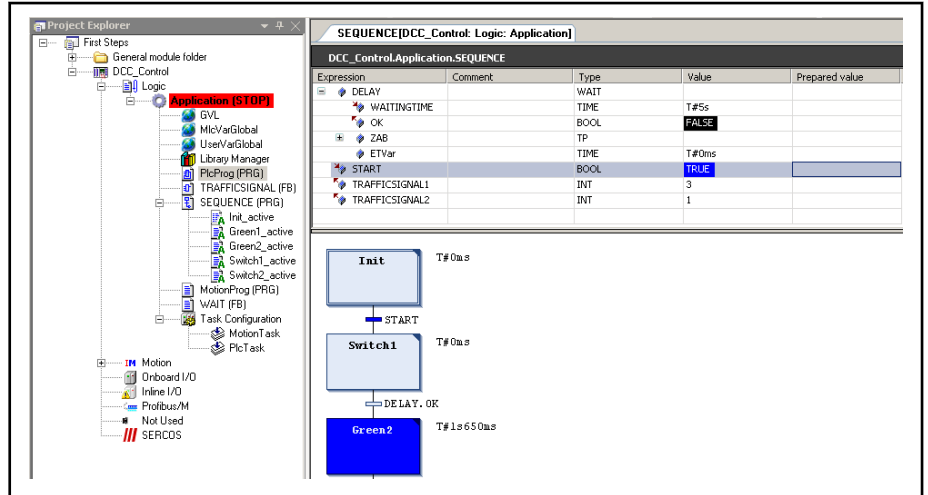


그림 4-51: 컨트롤에 로그인 - 응용 프로그램(중지)

**디버그(Debug) ▶ 다음 위치에서 로그아웃(Logout from)**을 선택하여 로그아웃합니다.

IndraWorks 가 오프라인 모드가 됩니다.



다시 로그인하면 마지막에 로그아웃했을 때의 값이 사용됩니다.

로그인하기 전에 이전 데이터가 삭제된 경우 다시 할당하여 시작할 수 있습니다(메뉴 항목 **만들기(Create) ▶ 정리(Clean)**).

## 4.6.7 프로그램 및 평선블럭 편집 – 2 부

### "ST(구조체언어)"로 "PlcProg" 프로그램 편집



"PlcProg" 프로그램의 구현은 다음 두 가지 프로그래밍 언어로 제공되었습니다.

- "CFC(연결구조언어)"(Continuous Function Chart (CFC))로 프로그래밍하려면 [CFC\(연결구조언어\)로 PlcProg 프로그램 편집, 60 페이지](#)를 참조하십시오.
- "ST(구조체언어)"(Structured Text (ST))로 프로그래밍하려면 아래로 계속 진행합니다.

#### "PlcProg"의 기능

두 신호등에 대한 신호 단계의 시간 시퀀스는 모두 "SEQUENCE" 평선블럭에서 구현되었습니다. PLC의 입력 및 출력은 제어하고 평가할 수 있습니다. 두 신호등의 입력 하나와 출력 여섯 개에 대한 입력 및 출력 변수는 "GVL(전역변수리스트)"(Global variable list (GVL))에서 만들었습니다. 이제 "PlcProg" 프로그램에서 이러한 변수를 사용할 수 있습니다. 응용 프로그램은 이전에 시뮬레이션 모드에서만 테스트했습니다. 컨트롤에서도 응용 프로그램을 테스트할 수 있습니다.

#### 인스턴스 선언

"PlcProg" 프로그램은 "TRAFFICSIGNAL" 유형의 "LIGHT1" 변수와 "TRAFFICSIGNAL" 유형의 "LIGHT2" 변수를 받습니다.

이는 "TRAFFICSIGNAL" 평선블럭을 인스턴스화합니다.

변수를 지정하려면 [변수 선언\(36 페이지\)](#)을 참조하십시오.

## 첫 단계 - IndraLogic

다음과 같이 LIGHT1: TRAFFICSIGNAL;  
및 LIGHT2: TRAFFICSIGNAL 변수를 만듭니다.

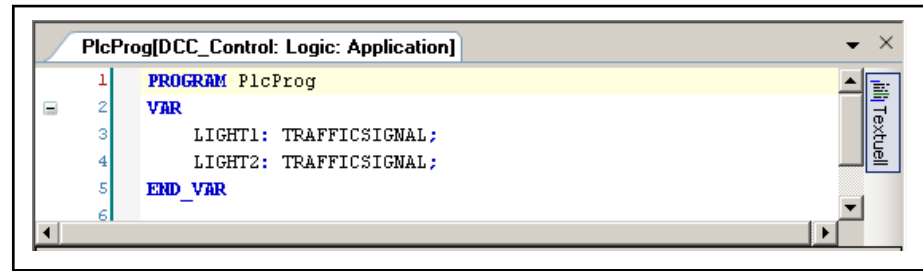


그림 4-52: "PlcProg" POU(프로그램 유닛)의 선언영역

## 본문 편집

"PlcProg" 프로그램은 "ST(구조체언어)" 프로그래밍 언어로 편집합니다. 첫 번째로 호출되는 평선블럭은 "SEQUENCE" 평선블럭입니다. "SEQUENCE" 프로그램의 "START" 입력에서 전역 변수 "ON"을 만듭니다.

그 다음에 "TRAFFICSIGNAL" 유형의 두 평선블럭 인스턴스 "LIGHT1" 및 "LIGHT2"가 호출됩니다.

"LIGHT1"의 "STATUS" 입력에서 "SEQUENCE" 프로그램의 출력 "TRAFFICSIGNAL1"을 만듭니다. "LIGHT1" 인스턴스의 세 가지 출력 "GREEN", "YELLOW" 및 "RED"가 전역 변수 "A1\_green", "A1\_yellow" 및 "A1\_red"에 기록됩니다.

"LIGHT1"과 같은 방법으로 평선블럭 인스턴스 "LIGHT2"에 대해 작업을 수행합니다.



"A1\_green"은 전역 변수이므로 정확히 "GVL.A1\_green"이라고 하거나 또는 간단히 ".A1\_green" 등으로도 부릅니다.

그러나 로컬로 선언된 "A1\_green"이 없으면 시스템에서는 그 대신 동일한 이름의 전역 변수를 사용합니다.

"PlcProg" 프로그램은 다음과 같이 표시됩니다.

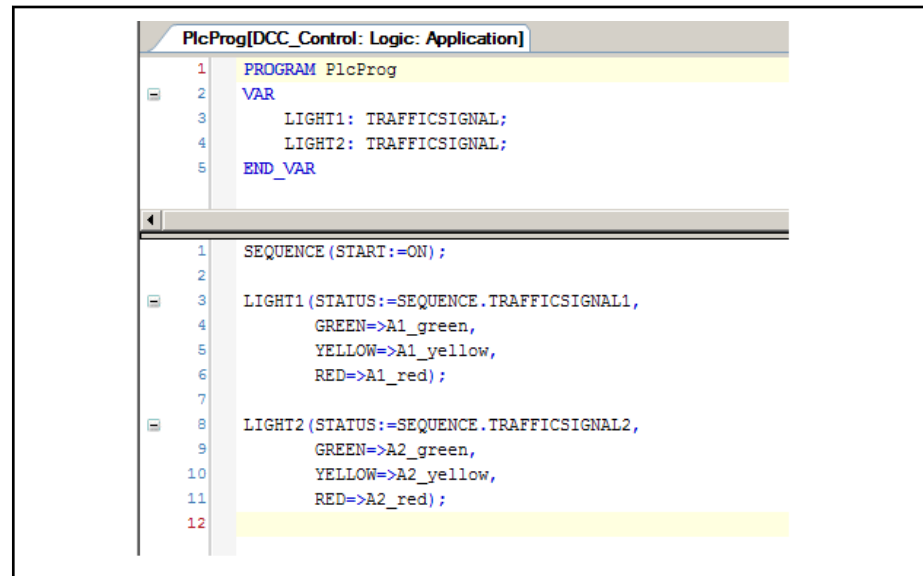


그림 4-53: 완료된 "PlcProg" 프로그램

## 시뮬레이션 모드에서 응용 프로그램 테스트(ST(구조체언어)로 작성한 PlcProg)

## "응용 프로그램" 컴파일

응용 프로그램을 컨트롤로 로드할 수 있으려면 먼저 "컴파일러"를 사용하여 실수 없이 응용 프로그램을 컴파일해야 합니다. 응용 프로그램을 컴파일하려면

## 첫 단계 - IndraLogic

**만들기(Create) ▶ 작성(Build)**을 선택합니다. 컴파일한 후에는 작업 창 아래의 "작업 목록"(Task list)에 "컴파일 완료 -- 오류 0 개, 경고 0 개: 다운로드 준비!"(Compiling completed -- 0 errors, 0 warnings: ready for download!)라는 메시지가 표시됩니다.



현재 활성 상태인 응용 프로그램만 컴파일됩니다.

여러 개의 응용 프로그램을 사용할 수 있는 경우 컴파일할 응용 프로그램을 활성으로 설정해야 합니다(상황에 맞는 메뉴 "응용 프로그램"(Application)활성 응용 프로그램 설정(Set Active Application)).

### 컨트롤 시뮬레이션

테스트에 사용할 수 있는 컨트롤이 없을 수 있으므로 컨트롤을 시뮬레이션할 수 있습니다. 컨트롤을 시뮬레이션하려면 "프로젝트 탐색기"(Project Explorer)에서 "DCC\_Control"을 클릭하고 마우스 오른쪽 버튼을 클릭하여 상황에 맞는 메뉴를 엽니다. **시뮬레이션(Simulation)**을 클릭하여 시뮬레이션 모드를 활성화합니다. 시뮬레이션이 활성화되면 "시뮬레이션"(Simulation)이 선택됩니다.

### 컨트롤에 로그인

컨트롤에 로그인한 후 응용 프로그램을 컨트롤로 로드해야 합니다. 컨트롤에 로그인하려면 메뉴에서 **디버그(Debug) ▶ 로그인(Login)**을 선택합니다.

그러면 응용 프로그램이 "중지" 상태가 됩니다.

### "응용 프로그램" 시작

응용 프로그램을 시작하려면 응용 프로그램을 마우스 오른쪽 버튼으로 클릭하고 상황에 맞는 메뉴에서 **시작(Start)**을 선택합니다. "응용 프로그램"이 이제 시뮬레이션으로 실행됩니다.

### 신호등을 켜고 신호 주기 관찰

응용 프로그램이 시작된 후 "LIGHT1"은 "녹색" 상태이고 "LIGHT2"는 "빨간색" 상태입니다. 신호 주기를 처리하려면 신호등을 켜야 합니다. 신호등을 켜려면 다음 단계를 진행합니다.

1. 프로젝트 탐색기에서 "GVL(전역변수리스트)"을 두 번 클릭하여 "전역변수리스트"(Global variable list)를 엽니다.

| GVL[DCC_Control: Logic: Application] |      |       |                |
|--------------------------------------|------|-------|----------------|
| DCC_Control.Application.GVL          |      |       |                |
| Expression                           | Type | Value | Prepared value |
| ON                                   | BOOL | FALSE |                |
| A1_green                             | BOOL | TRUE  |                |
| A1_yellow                            | BOOL | FALSE |                |
| A1_red                               | BOOL | FALSE |                |
| A2_green                             | BOOL | FALSE |                |
| A2_yellow                            | BOOL | FALSE |                |
| A2_red                               | BOOL | TRUE  |                |

그림 4-54: "GVL(전역변수리스트)" 온라인

2. "준비된 값"(Prepared value) 열에서 "ON" 변수 옆을 클릭하여 "TRUE"를 표시합니다.
3. 컨트롤에 값을 쓰려면 **디버그(Debug) ▶ 값 쓰기(Write values)**를 클릭합니다.

신호 주기가 처리됩니다. "PlcProg" 평선블럭에서 신호 주기를 볼 수 있습니다.

## 첫 단계 - IndraLogic

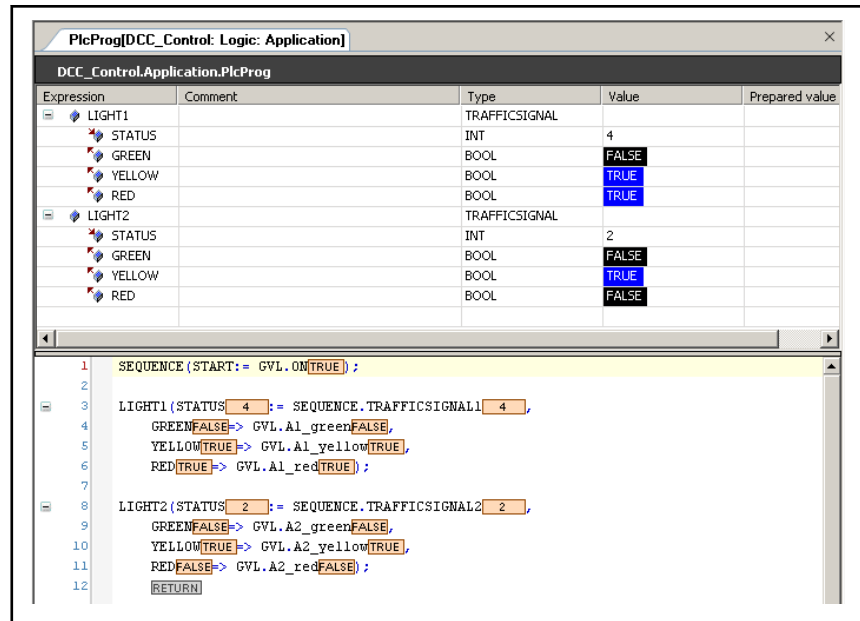


그림 4-55: "PlcProg" 프로그램 온라인

**응용 프로그램 중지** 메뉴 모음의 디버그(Debug) ▶ 중지(Stop)를 통해 PLC 프로그램을 중지합니다.

**컨트롤에서 로그아웃** 컨트롤에서 로그아웃하려면 디버그(Debug) ▶ 로그아웃(Logout)을 선택합니다.

**"CFC(연결구조언어)"로 "PlcProg" 프로그램 편집**

"PlcProg" 프로그램의 구현은 다음 두 가지 프로그래밍 언어로 제공되었습니다.

- "ST(구조체언어)"(Structured Text (ST))로 프로그래밍할 경우 ST(구조체언어)로 PlcProg 프로그램 편집, 57 페이지를 참조하십시오.
- "CFC(연결구조언어)"(Continuous Function Chart (CFC))로 프로그래밍할 경우에는 아래로 계속 진행합니다.

**"PlcProg"의 기능** 두 신호등에 대한 신호 단계의 시간 시퀀스는 모두 "SEQUENCE" 평선블럭에서 구현되었습니다. PLC의 입력 및 출력은 제어하고 평가할 수 있습니다. 두 신호등의 입력 하나와 출력 여섯 개에 대한 입력 및 출력 변수는 "GVL(전역변수리스트)"(Global variable list (GVL))에서 만들었습니다. 이제 "PlcProg" POU(프로그램 유닛)에서 이러한 변수를 사용할 수 있습니다. 응용 프로그램은 이전에 시뮬레이션 모드에서만 테스트했습니다. 컨트롤에서도 응용 프로그램을 테스트할 수 있습니다.

**인스턴스 선언** "PlcProg" 프로그램은 "TRAFFICSIGNAL" 유형의 "LIGHT1" 인스턴스와 "TRAFFICSIGNAL" 유형의 "LIGHT2" 인스턴스를 받습니다.

이는 "TRAFFICSIGNAL" 평선블럭을 인스턴스화합니다.

다음과 같이 LIGHT1: TRAFFICSIGNAL; 및 LIGHT2: TRAFFICSIGNAL 변수를 만듭니다.

첫 단계 - IndraLogic

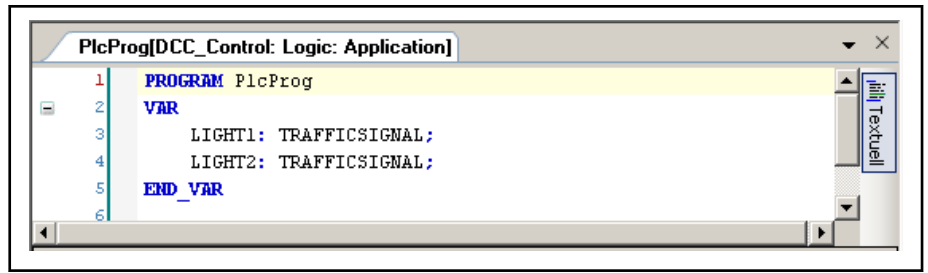


그림 4-56: "PlcProg" POU(프로그램 유닛)의 선언영역

본문 편집

"PlcProg" POU(프로그램 유닛)는 "CFC(연결구조언어)" 프로그래밍 언어로 편집합니다. 첫 번째로 호출되는 평선블럭은 "SEQUENCE" 평선블럭입니다.

1. "도구 상자"(ToolBox) 창에서 "상자"(Box)를 클릭하여 선택합니다.

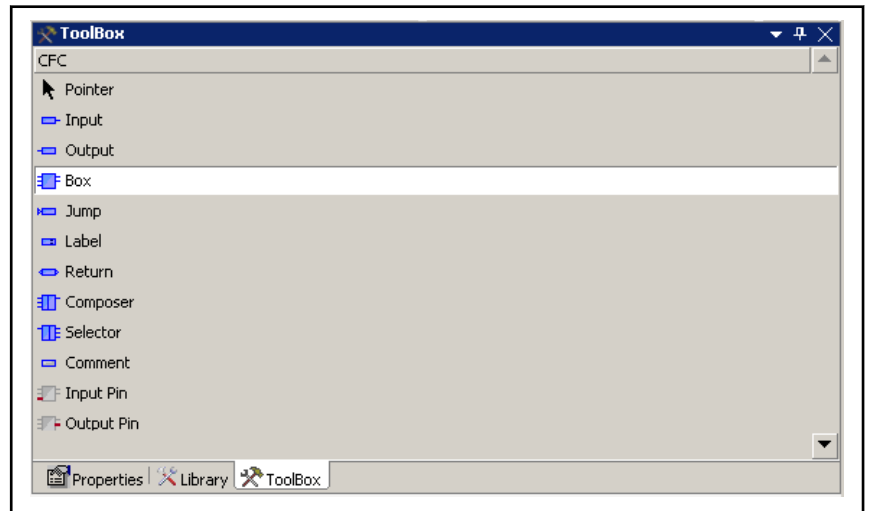


그림 4-57: 도구 - 평선블럭 요소

2. 마우스를 "PlcProg" 편집기의 본문에 놓고 마우스 왼쪽 버튼을 클릭하여 여기에 평선블럭을 추가합니다.
3. 평선블럭의 이름인 "???"를 클릭하고 평선블럭 이름으로 "SEQUENCE"를 입력합니다.

FB(평선블럭)의 모양은 다음과 같습니다.

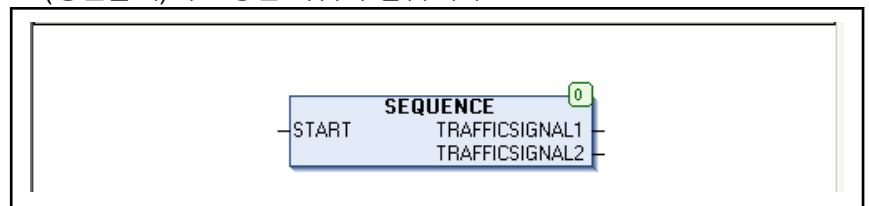


그림 4-58: SEQUENCE 평선블럭

"SEQUENCE" 평선블럭의 "TRAFFICSIGNAL1" 출력에 "TRAFFICSIGNAL" 기능과 함께 "LIGHT1" 평선블럭을 추가합니다.

1. "도구 상자"(ToolBox) 창에서 "블럭(상자)"(Block (Box)) 요소를 클릭하여 선택합니다.
2. 마우스를 "PlcProg" 편집기의 본문에 놓고 마우스 왼쪽 버튼을 클릭하여 여기에 평선블럭을 추가합니다.
3. 평선블럭의 이름인 "???"를 클릭하고 평선블럭의 유형 이름으로 "TRAFFICSIGNAL"을 지정합니다.

## 첫 단계 - IndraLogic

4. "???"에서 "TRAFFICSIGNAL" 평선블럭의 인스턴스 이름으로 "LIGHT1"을 지정합니다.

FB(평선블럭)의 모양은 다음과 같습니다.

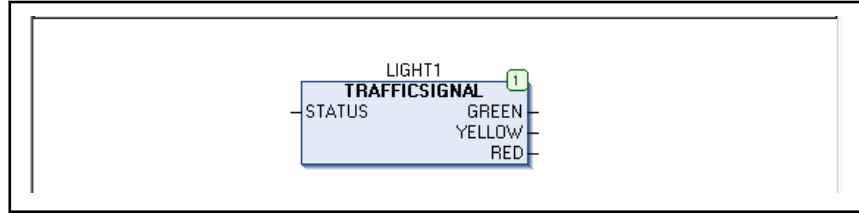


그림 4-59: TRAFFICSIGNAL - LIGHT1 평선블럭

다음으로는 "SEQUENCE" 블럭의 "TRAFFICSIGNAL2" 출력에 "TRAFFICSIGNAL" 기능과 함께 "LIGHT2" 평선블럭을 추가합니다.

1. "도구 상자"(ToolBox) 창에서 "블럭(상자)"(Block (Box)) 요소를 클릭하여 선택합니다.
2. 마우스를 "PlcProg" 편집기의 본문에 놓고 마우스 왼쪽 버튼을 클릭하여 여기에 평선블럭을 추가합니다.
3. 평선블럭의 이름인 "???"를 클릭하고 평선블럭 이름으로 "TRAFFICSIGNAL"을 입력합니다.
4. "???"에서 "TRAFFICSIGNAL" 평선블럭의 인스턴스 이름으로 "LIGHT2"를 지정합니다.

FB(평선블럭)의 모양은 다음과 같습니다.

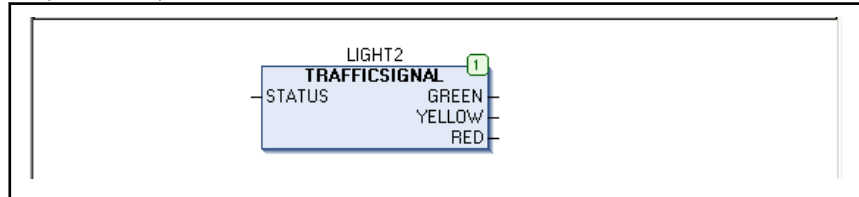


그림 4-60: TRAFFICSIGNAL - LIGHT2 평선블럭

그런 후 다음과 같이 "입력"(Input) 및 "출력"(Output) 요소를 추가합니다.

1. "도구 상자"(ToolBox) 창에서 "입력"(Input) 요소를 클릭하여 선택합니다.
2. 마우스를 "PlcProg" 편집기의 본문에 놓고 마우스 왼쪽 버튼을 클릭하여 여기에 입력을 추가합니다.
3. 입력 이름인 "???"를 클릭하고 입력 이름으로 "ON"을 입력합니다.
4. "도구 상자"(ToolBox) 창에서 "출력"(Output) 요소를 클릭하여 선택합니다.
5. 마우스를 "PlcProg" 편집기의 본문에 놓고 마우스 왼쪽 버튼을 클릭하여 여기에 출력을 추가합니다.
6. "PlcProg"에 출력을 다섯 개 더 추가합니다.
7. 출력 이름을 "A1\_green", "A1\_yellow", "A1\_red", "A2\_green", "A2\_yellow" 및 "A2\_red"로 지정합니다.



"A1\_green"은 전역 변수이므로 정확히 "GVL.A1\_green"이라고 하거나 또는 간단히 ".A1\_green" 등으로도 부릅니다.

그러나 로컬로 선언된 "A1\_green"이 없으면 시스템에서는 그 대신 동일한 이름의 전역 변수를 사용합니다.

네트워크를 완료하려면 다음 단계를 진행합니다.

## 첫 단계 - IndraLogic

1. "ON" 요소의 출력을 마우스 왼쪽 버튼으로 클릭하고 마우스 키를 누르고 있습니다. 두 요소가 서로 연결되도록 마우스 포인터를 "SEQUENCE" 평션블럭의 "START" 입력으로 이동합니다.
2. "SEQUENCE" 요소의 "TRAFFICSIGNAL1" 출력을 마우스 왼쪽 버튼으로 클릭하고 마우스 키를 누르고 있습니다. 두 요소가 서로 연결되도록 마우스 포인터를 "TRAFFICSIGNAL" 평션블럭의 "STATUS" 입력으로 이동합니다.
3. "SEQUENCE" 요소의 "TRAFFICSIGNAL2" 출력을 마우스 왼쪽 버튼으로 클릭하고 마우스 키를 누르고 있습니다. 두 요소가 서로 연결되도록 마우스 포인터를 "LIGHT2" 평션블럭의 "STATUS" 입력으로 이동합니다.
4. "LIGHT1" 요소의 "GREEN" 출력을 마우스 왼쪽 버튼으로 클릭하고 마우스 키를 누르고 있습니다. 두 요소가 서로 연결되도록 마우스 포인터를 출력의 "A1\_green" 입력으로 이동합니다.
5. 필요에 따라 다른 출력에 대해 마지막 단계를 반복합니다.

"PlcProg" 프로그램은 다음과 같이 표시됩니다.

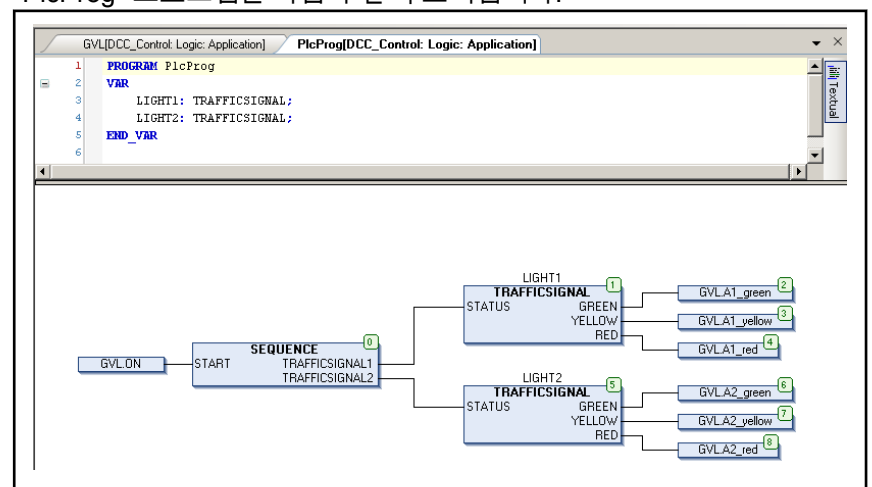


그림 4-61: 완료된 "PlcProg" POU(프로그램 유닛)



CFC(연결구조언어) 메뉴를 사용하여 "입력"(Input), "출력"(Output) 및 "블럭"(Block) 등의 요소를 추가할 수도 있습니다.

## 시뮬레이션 모드에서 응용 프로그램 테스트(CFC(연결구조언어)로 작성한 PlcProg)

### "응용 프로그램" 컴파일

응용 프로그램을 컨트롤로 로드할 수 있으려면 먼저 "컴파일러"를 사용하여 실행 없이 응용 프로그램을 컴파일해야 합니다. 응용 프로그램을 컴파일하려면 **만들기(Create) > 응용 프로그램 작성(Build Application)**을 선택합니다. 컴파일한 후에는 작업 창 아래의 "작업 목록"(Task list)에 "컴파일 완료 -- 오류 0 개, 경고 0 개: 다운로드 준비!" (Compiling completed -- 0 errors, 0 warnings: ready for download!)라는 메시지가 표시됩니다.



현재 활성 상태인 응용 프로그램만 컴파일됩니다.

여러 개의 응용 프로그램을 사용할 수 있는 경우 컴파일할 응용 프로그램을 활성으로 설정해야 합니다(상황에 맞는 메뉴 "응용 프로그램"(Application) **활성 응용 프로그램 설정(Set Active Application)**).

### 컨트롤 시뮬레이션

테스트에 사용할 수 있는 컨트롤이 없을 수 있으므로 컨트롤을 시뮬레이션할 수 있습니다. 컨트롤을 시뮬레이션하려면 "프로젝트 탐색기"(Project Explorer)에서 "DCC\_Control"을 클릭하고 마우스 오른쪽 버튼을 클릭하여 상황에 맞는 메뉴를 엽니다. **시뮬레이션(Simulation)**을 클릭하여 시뮬레이션 모드를 활성화합니다. 시뮬레이션이 활성화되면 "시뮬레이션"(Simulation)이 선택됩니다.

## 첫 단계 - IndraLogic

## 컨트롤에 로그인

컨트롤에 로그인한 후 응용 프로그램을 컨트롤로 로드해야 합니다. 컨트롤에 로그인하려면 메뉴에서 **디버그(Debug) ▶ 로그인(Login)**을 선택합니다. 응용 프로그램을 온라인으로 변경하거나 다시 컴파일하여 새 응용 프로그램을 컨트롤로 로드할 수 있습니다.

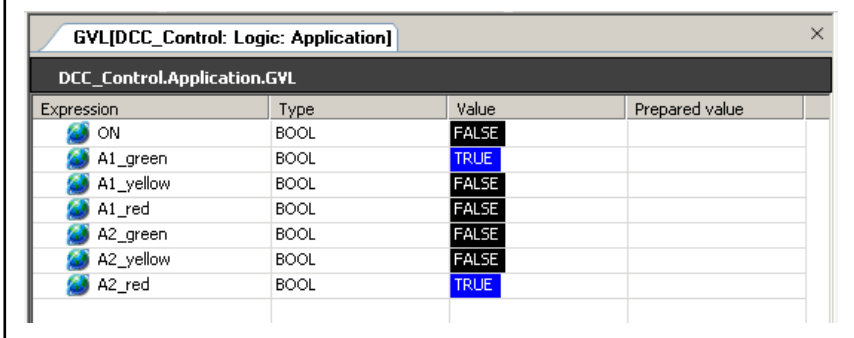
## "응용 프로그램" 시작

응용 프로그램을 시작하려면 응용 프로그램을 마우스 오른쪽 버튼으로 클릭하고 상황에 맞는 메뉴에서 **시작(Start)**을 선택합니다. "응용 프로그램"이 이제 시뮬레이션으로 실행됩니다.

## 신호등을 켜고 신호 주기 관찰

응용 프로그램이 시작된 후 "LIGHT1"은 "녹색" 상태이고 "LIGHT2"는 "빨간색" 상태입니다. 신호 주기를 처리하려면 신호등을 켜야 합니다. 신호등을 켜려면 다음 단계를 진행합니다.

1. 프로젝트 탐색기에서 "GVL(전역변수리스트)"을 두 번 클릭하여 "전역변수리스트"(Global variable list)를 엽니다.



| Expression | Type | Value | Prepared value |
|------------|------|-------|----------------|
| ON         | BOOL | FALSE |                |
| A1_green   | BOOL | TRUE  |                |
| A1_yellow  | BOOL | FALSE |                |
| A1_red     | BOOL | FALSE |                |
| A2_green   | BOOL | FALSE |                |
| A2_yellow  | BOOL | FALSE |                |
| A2_red     | BOOL | TRUE  |                |

그림 4-62: "GVL(전역변수리스트)" 온라인

2. "준비된 값"(Prepared value) 열에서 "ON" 변수 옆을 클릭하여 "TRUE"를 표시합니다.
3. 컨트롤에 값을 쓰려면 **디버그(Debug) ▶ 값 쓰기(Write values)**를 클릭합니다.

신호 주기가 처리됩니다. "PlcProg" 평션블럭에서 신호 주기를 볼 수 있습니다.

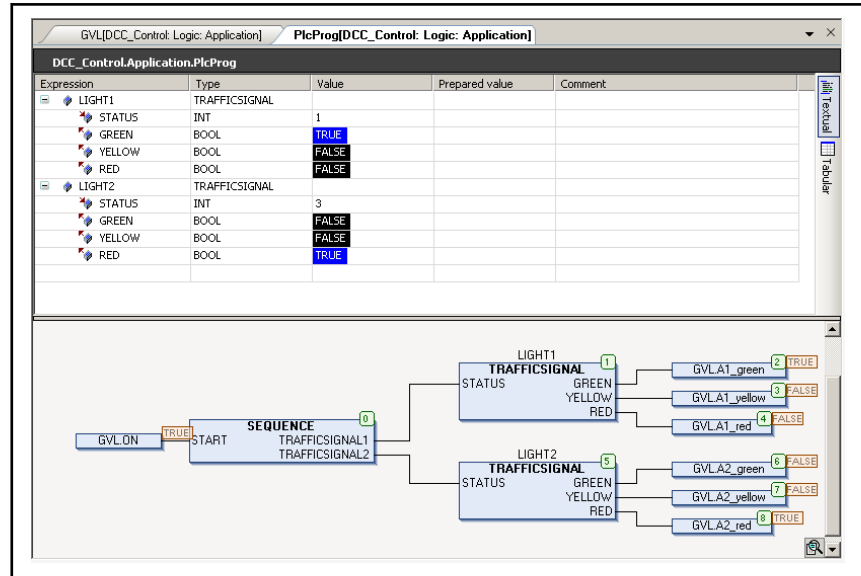


그림 4-63: "PlcProg" 프로그램 온라인

## 응용 프로그램 중지

메뉴 모음의 **디버그(Debug) ▶ 중지(Stop)**를 통해 PLC 프로그램을 중지합니다.

컨트롤에서 로그아웃      컨트롤에서 로그아웃하려면 디버그(Debug) ▶ 로그아웃(Logout)을 선택합니다.

## 4.6.8 컨트롤에서 응용 프로그램 테스트

시뮬레이션 모드 비활성화      컨트롤을 사용할 수 있는 경우 컨트롤에서 응용 프로그램을 테스트해야 합니다.

이렇게 하려면 [프로젝트의 통신 설정\(27 페이지\)](#)이 필요한 컨트롤과 일치해야 합니다.

시뮬레이션 모드를 끝내려면 "프로젝트 탐색기"에서 "DCC\_Control" 컨트롤을 마우스 오른쪽 버튼으로 클릭하여 상황에 맞는 메뉴를 엽니다. **시뮬레이션(Simulation)**을 클릭하여 시뮬레이션 모드를 비활성화합니다. 시뮬레이션이 더 이상 활성 상태가 아니면 "시뮬레이션"(Simulation)이 선택 취소됩니다.

"응용 프로그램" 컴파일

응용 프로그램을 컨트롤로 로드하려면 먼저 "컴파일러"를 사용하여 오류 없이 응용 프로그램을 컴파일해야 합니다. 응용 프로그램을 컴파일하려면 **만들기(Create) ▶ 응용 프로그램 작성(Build Application)**을 선택합니다. 컴파일한 후에는 작업 창 아래의 "작업 목록"(Task list)에 "컴파일 완료 -- 오류 0 개, 경고 0 개: 다운로드 준비! "(Compiling completed -- 0 errors, 0 warnings: ready for download!)라는 메시지가 표시됩니다.



현재 활성 상태인 응용 프로그램만 컴파일됩니다.

여러 개의 응용 프로그램을 사용할 수 있는 경우 컴파일할 응용 프로그램을 활성으로 설정해야 합니다(상황에 맞는 메뉴 "응용 프로그램"(Application) **활성 응용 프로그램 설정(Set Active Application)**).

컨트롤에 로그인

컨트롤에 로그인한 후 응용 프로그램을 컨트롤로 로드해야 합니다. 컨트롤에 로그인하려면 메뉴에서 **디버그(Debug) ▶ 로그인(Login)**을 선택합니다.

그러면 응용 프로그램이 "중지" 상태가 됩니다.

"응용 프로그램" 시작

응용 프로그램을 시작하려면 응용 프로그램을 마우스 오른쪽 버튼으로 클릭하고 상황에 맞는 메뉴에서 **시작(Start)**을 선택합니다. 응용 프로그램이 컨트롤에서 실행됩니다.

부트 응용 프로그램 생성

메뉴 모음을 사용하여 부트 프로젝트를 생성하려면 **디버그(Debug) ▶ 부트 응용 프로그램 생성(Generate boot application)**을 선택합니다. 이 명령을 온라인으로 실행하면 다음에 시작할 때 자동으로 로드되도록 컴파일된 프로젝트가 컨트롤에 저장됩니다.

신호등을 켜고 신호 주기 관찰

응용 프로그램이 시작된 후 "교통 신호 1"은 "녹색" 상태이고 "교통 신호 2"는 "빨간색" 상태입니다. 신호 주기를 처리하려면 신호등을 켜야 합니다. 신호등을 켜려면 다음 단계를 진행합니다.

1. 프로젝트 탐색기에서 "GVL(전역변수리스트)"을 두 번 클릭하여 전역변수리스트를 엽니다.

| Expression | Type | Value | Prepared value |
|------------|------|-------|----------------|
| ON         | BOOL | FALSE |                |
| A1_green   | BOOL | TRUE  |                |
| A1_yellow  | BOOL | FALSE |                |
| A1_red     | BOOL | FALSE |                |
| A2_green   | BOOL | FALSE |                |
| A2_yellow  | BOOL | FALSE |                |
| A2_red     | BOOL | TRUE  |                |

그림 4-64: "GVL(전역변수리스트)" 온라인

## 첫 단계 - IndraLogic

2. "준비된 값"(Prepared value) 열에서 "ON" 변수 옆을 클릭하여 "TRUE"를 표시합니다.
3. 컨트롤에 값을 쓰려면 디버그(Debug) ▶ 값 쓰기(Write values)를 클릭합니다.

신호 주기가 처리됩니다.

그런 다음 개별 프로그램 POU(프로그램 유닛)(PlcProg, SEQUENCE) 및 GVL(전역변수리스트)에서 교통 신호 주기를 처리할 수 있습니다.

TRAFFICSIGNAL 평선블럭을 엽니다. "PlcProg.LIGHT1" 또는 "PlcProg.LIGHT2" 중에서 선택할 수 있는 선택 창이 먼저 나타납니다.

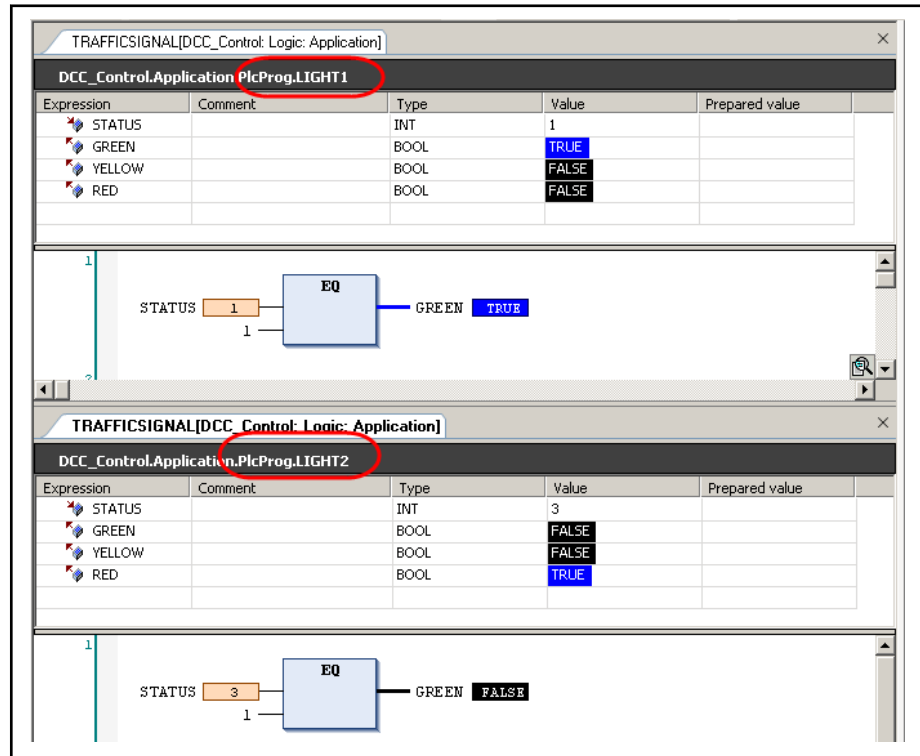


그림 4-65: TRAFFICSIGNAL의 평선블럭 인스턴스 두 개에 대한 상태 표시

#### 실제 입력 및 출력의 컨트롤

IndraMotion MLC/IndraLogic XLC L45 및 L65 컨트롤에서는 %QX0.0 ... %QX0.7 주소에 할당된 온보드 출력이 표시됩니다(매핑). 이러한 출력에서 신호 주기를 따를 수도 있습니다.

#### 응용 프로그램 중지

메뉴 모음의 디버그(Debug) ▶ 중지(Stop)를 통해 PLC 프로그램을 중지합니다.

#### 컨트롤에서 로그아웃

컨트롤에서 로그아웃하려면 디버그(Debug) ▶ 로그아웃(Logout)을 선택합니다.

## 4.7 PLC 프로그램의 시각화 만들기

### 4.7.1 IndraLogic의 시각화

IndraLogic 시각화를 사용하면 동적 절차의 시각적 묘사를 쉽고 빠르게 만들 수 있습니다. 정의된 간격마다 변수의 상태가 쿼리되고 각 상태에 따라 해당하는 그래픽 요소가 표시됩니다. 여기에서는 "PLC 프로그램 만들기"(29 페이지)에서 설명한 PLC 프로그램을 시각화합니다. 프로그래밍된 신호등에 대한 다음 두 개의 신호와 ON 스위치를 시각화하고 해당하는 상태를 표시합니다.

### 4.7.2 새 시각화 만들기

새 시각화를 만들려면 다음 단계를 진행합니다.

## 첫 단계 - IndraLogic

1. "라이브러리(PLC 개체/VI 논리적 개체)"에서 "프로젝트 탐색기"(Project Explorer)의 응용 프로그램으로 "시각화"(Visualization) 요소를 끌어다 놓아 응용 프로그램에 시각화를 추가합니다.

다음 창이 열립니다.

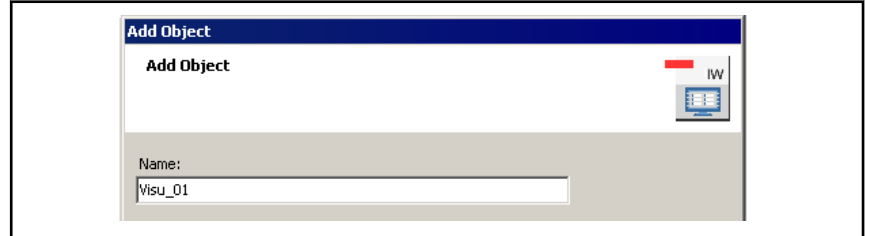


그림 4-66: 응용 프로그램에 시각화 추가

2. 시각화 이름을 원하는 대로 지정하고 "완료"(Finish)를 클릭하여 확인합니다.

시각화 관리자의 인스턴스와 함께 시각화가 추가됩니다.

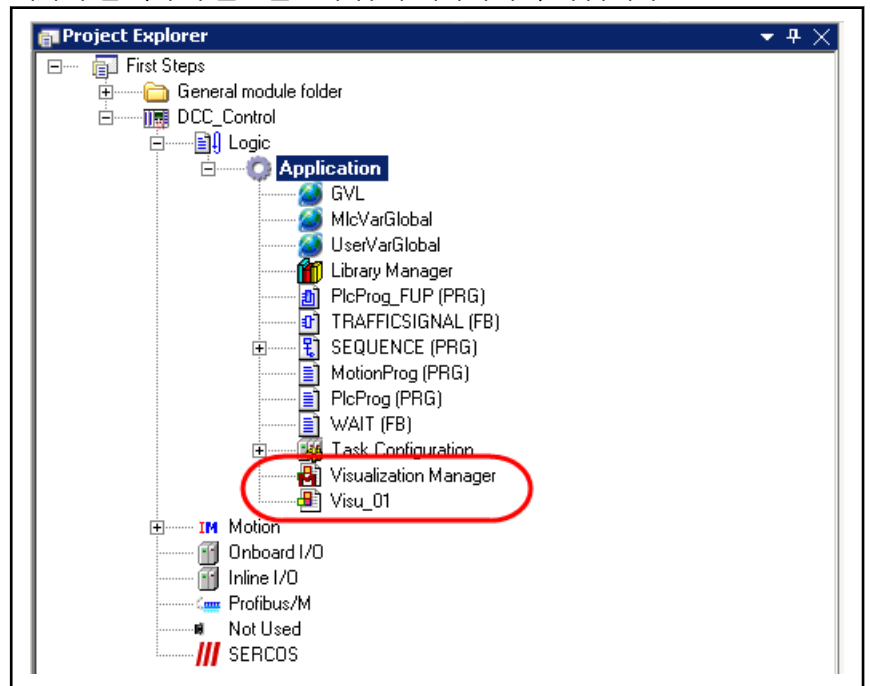


그림 4-67: 응용 프로그램에 추가된 시각화

여기에서 필요하지 않은 요소도 "일반 모듈"(General module) 폴더에 추가로 생성됩니다.

### 4.7.3 시각화 편집

#### 첫 번째 요소 "신호 램프" 추가 및 편집

**시각화 편집기 열기** 프로젝트 탐색기에서 시각화를 두 번 클릭하여 시각화 편집기를 엽니다.

왼쪽 위에 (X, Y) = (0, 0) 위치가 있는 "도면 시트"가 제공됩니다.

도구(Tools) ▶ 옵션(Options) ▶ IndraLogic 2G ▶ 시각화(Visualization)를 통해 그리드를 표시하고 활성화할 수 있습니다.

도구 상자에는 다음 세 가지 폴더가 제공됩니다.

- 기본: 일반 시각화 개체
- 창 컨트롤: 대화 상자를 디자인하는 데 필요한 요소

## 첫 단계 - IndraLogic

- 복합 컨트롤: 복합 요소(예: "측정기"), 트레이스 객체
- 타원 요소 추가** "교통 신호"의 첫 번째 요소인 "조명"(Light)을 시각화에 추가합니다. 시각화에 추가할 수 있는 요소는 "도구 상자"(ToolBox) 창에 있습니다. 다음 그림을 참조하십시오.

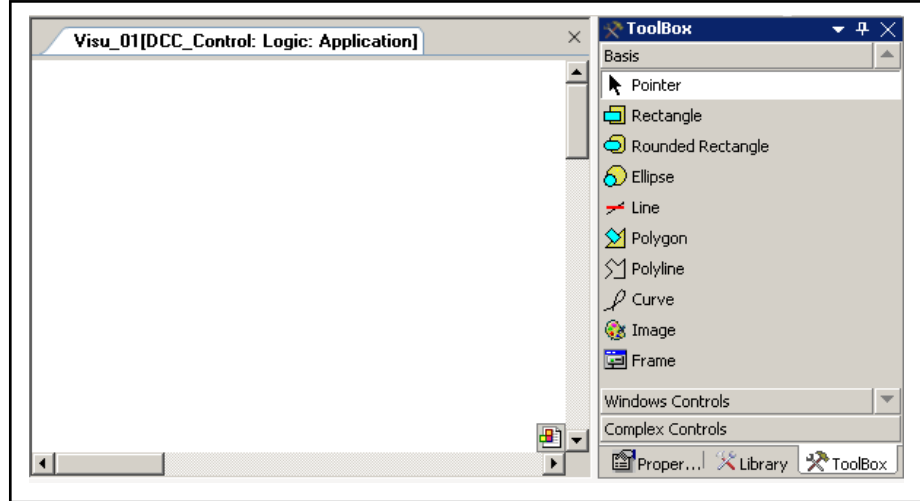


그림 4-68: "도구 상자"(ToolBox) 창

첫 번째 요소인 "조명"(Light)을 추가하려면 다음 단계를 진행합니다.

- "도구 상자"(ToolBox) 창에서 "타원"(Ellipse) 요소를 마우스 왼쪽 버튼으로 클릭하여 선택합니다.
- 마우스를 시각화 창으로 이동한 후 다시 마우스 왼쪽 버튼을 클릭합니다. "타원" 요소가 시각화 편집기에 추가됩니다.

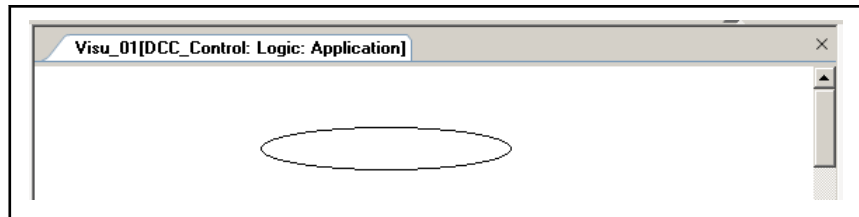


그림 4-69: 시각화 편집기 - 첫 번째 타원

**타원(평선블럭)의 모양 변경**

여기에서는 추가한 "타원"(평선블럭)의 모양을 "신호등"(평선블럭)의 모양과 일치하도록 변경합니다.

"타원"(평선블럭)의 모양과 위치를 변경하려면 다음 단계를 진행합니다.

- "타원"을 마우스 왼쪽 버튼으로 클릭하여 "타원"을 강조합니다.
- "속성"(Properties) 탭을 마우스 왼쪽 버튼으로 클릭하여 "속성"(Properties) 창을 엽니다.

첫 단계 - IndraLogic

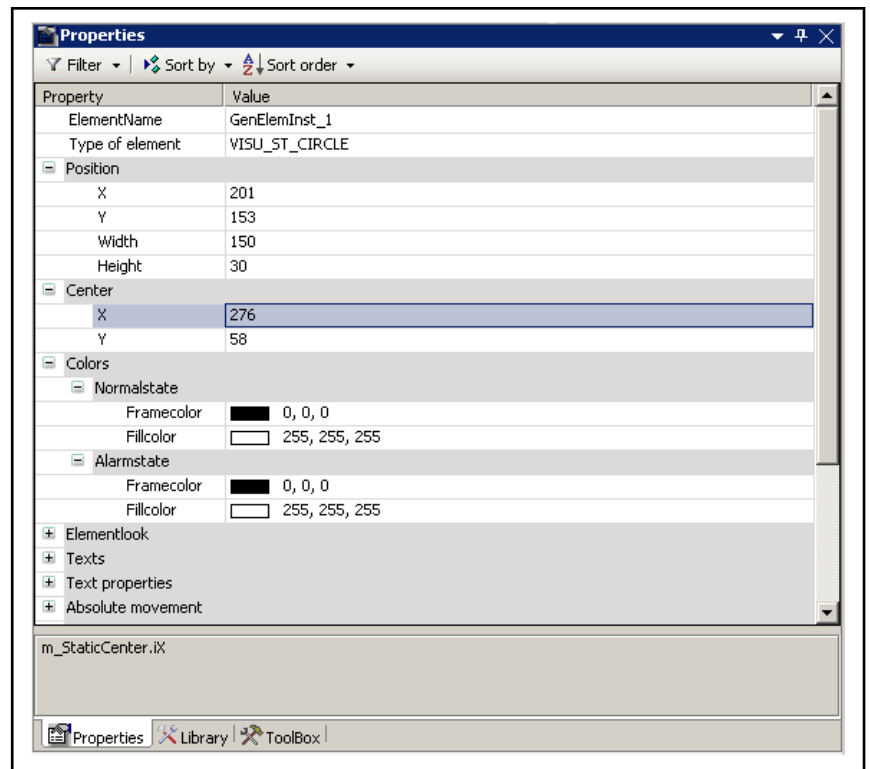


그림 4-70: 타원의 "속성"(Properties) 창

- (X, Y) 위치 및 크기(너비, 높이)를 변경하려면 "위치"(Position) 속성 아래에서 다음을 선택합니다.

"X" = "60"

"Y" = "150"

"너비"(Width) = "30"

"높이"(Height) = "30"

- "타원"의 중심점을 변경하려면 다음과 같이 "중심"(Center) 아래에서 (X, Y)를 선택합니다.

"X" = "75"

"Y" = "165"

#### 신호등의 속성 변경

여기에서는 신호등의 채우기 색을 변경합니다. 신호등은 비활성 상태일 때는 회색이고 활성 상태일 때는 빨간색이어야 합니다.

채우기 색을 변경하려면 다음 단계를 진행합니다.

- 정상 조건(정상 상태)의 채우기 색을 변경하려면 "색"(Colors) 속성의 "정상 상태"(Normal state)에 있는 "채우기 색"(Fill color)에서 "..." 버튼을 클릭합니다.

"색"(Color) 창이 열립니다.

- "색"(Color) 창에서 "채우기 색"(Fill Colors)을 클릭합니다. "은색"(Silver)을 선택합니다.

- 경보 상태의 채우기 색을 변경하려면 "색"(Colors) 속성의 "경보 상태"(Alarm state)에 있는 "채우기 색"(Fill color)에서 "..." 버튼을 클릭합니다.

"색"(Color) 창이 열립니다.

## 첫 단계 - IndraLogic

4. "색"(Color) 창에서 "채우기 색"(FillColors)을 클릭합니다. "빨간색"(Red)을 선택합니다.

정상 조건(정상 상태)에서 신호등은 회색입니다. 신호등은 "빨간색" 상태일 때 빨간색으로 켜져야 합니다. "PlcProg" POU(프로그램 유닛)에서 "GVL(전역변수리스트)"의 "A1\_red" 변수가 "TRUE"로 설정될 때 신호등이 빨간색으로 켜집니다. 신호등 상태를 동적으로 만들려면 신호등 속성에 신호 색을 변경하기 위한 변수를 입력해야 합니다.

다음과 같이 진행합니다.

1. 신호 색을 변경하기 위한 변수를 시각화에 전달하려면 "색 변수"(Color variables) 아래의 "색 전환"(ToggleColor)에서 "값"(Value) 열을 두 번 클릭합니다.
2. "..." 버튼을 클릭합니다.  
"입력 보조툴"(Input Assistance) 창이 열립니다.
3. "입력 보조툴"(Input Assistant) 창의 "요소"(Elements) 아래에 "응용 프로그램"(Application), "GVL(전역변수리스트)"이 표시될 수 있습니다.
4. "GVL(전역변수리스트) 변수"(GVL variables) 아래에서 "A1\_red" 변수를 선택합니다.

"GVL(전역변수리스트) 변수"가 "색 전환"(ToggleColor)에 추가됩니다.

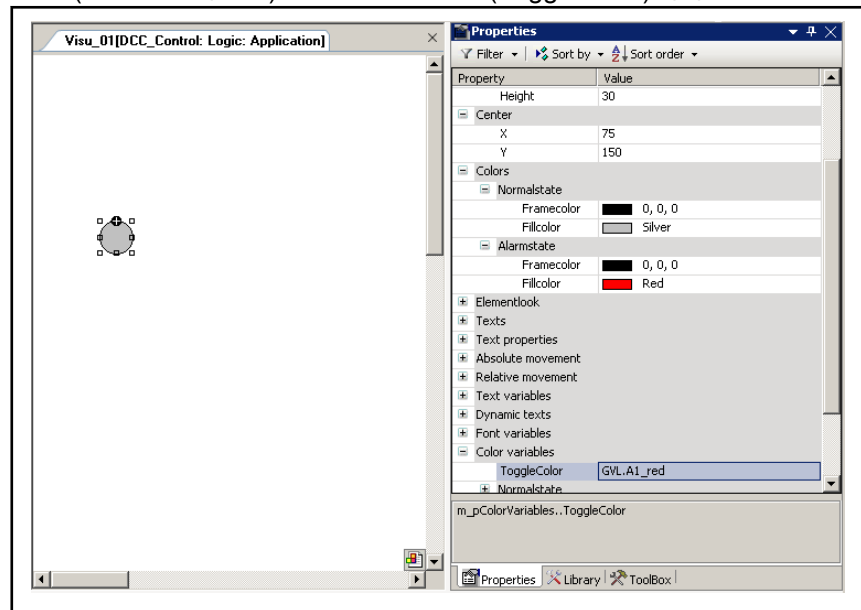


그림 4-71: 속성(Properties) - 색 전환(ToggleColor)

## "신호 램프" 요소 복사, 추가 및 편집

### "신호등" 요소 복사 및 삽입

이제 첫 번째 신호등의 "빨간색" 신호가 완료되었습니다. 첫 번째 신호등에는 신호 색이 "노란색" 및 "녹색"인 신호등만 필요합니다. 두 개의 다른 신호등은 첫 번째 신호등과 단계가 동일하므로 가장 쉬운 방법은 첫 번째 신호등을 복사하는 것입니다.

신호등을 복사하여 삽입하려면 다음 단계를 진행합니다.

1. 신호등을 클릭하여 강조합니다.
2. 마우스 오른쪽 버튼을 사용하여 상황에 맞는 메뉴를 열고 복사(Copy)를 선택합니다.
3. 시각화 영역을 클릭하고 상황에 맞는 메뉴에서 삽입(Insert)을 선택합니다.

첫 단계 - IndraLogic

4. 신호등을 더 추가하려면 시각화 편집기에서 비어 있는 영역을 다시 클릭한 다음 상황에 맞는 메뉴에서 **삽입(Insert)**을 선택합니다.
5. 처음에 추가한 신호등을 마우스 왼쪽 버튼으로 클릭하여 강조합니다.

**"신호등" 요소의 위치 변경**

이제 방금 추가한 신호등 두 개의 위치를 변경해야 합니다.

신호등의 위치를 변경하려면 다음 단계를 진행합니다.

1. 처음에 추가한 신호등을 마우스 왼쪽 버튼으로 클릭하여 강조합니다.
2. "속성"(Properties) 창의 "위치"(Position)에서 (X, Y)에 대해 (60, 195)를 선택하여 위치를 변경합니다.
3. 중심점을 변경하려면 "중심"(Center) 아래에서 (X, Y)에 대해 (75, 210)을 선택합니다.
4. 두 번째로 추가한 신호등을 마우스 왼쪽 버튼으로 클릭하여 강조합니다.
5. "속성"(Properties) 창의 "위치"(Position)에서 (X, Y)에 대해 (60, 240)를 선택하여 위치를 변경합니다.
6. 중심점을 변경하려면 "중심"(Center) 아래에서 (X, Y)에 대해 (75, 255)을 선택합니다.

**"신호등" 요소의 속성 변경**

여기에서는 두 신호등의 "경보 상태"(Alarm state)에 대한 새로운 색을 할당합니다.

다음과 같이 진행합니다.

1. 가운데에 있는 신호등을 강조합니다.
2. 경보 상태의 채우기 색을 변경하려면 "색"(Colors) 속성의 "경보 상태"(Alarm state)에 있는 "채우기 색"(Fill color)에서 "..." 버튼을 클릭합니다.
3. "색"(Color) 창이 열립니다.
4. "색"(Color) 창에서 "채우기 색"(FillColors)을 클릭합니다. "노란색"(Yellow)을 선택합니다.
5. 다른 신호등에 대해 1 - 3 단계를 반복합니다.
6. "색"(Color) 창에서 "채우기 색"(FillColors)을 클릭합니다. "녹색"(Green)을 선택합니다.

다양한 신호 색을 처리하기 위한 변수를 조정해야 합니다.

다음과 같이 진행합니다.

1. 가운데에 있는 신호등을 강조합니다.
2. 신호 색을 변경하기 위한 변수를 시각화에 전달하려면 "색 변수"(Color variables) 아래의 "색 전환"(ToggleColor)에서 "값"(Value) 열을 두 번 클릭합니다.

"..." 버튼을 클릭합니다.

"입력 보조툴"(Input Assistance) 대화 상자가 열립니다.

3. "입력 보조툴"(Input Assistant) 창의 "요소"(Elements) 아래에 "응용 프로그램"(Application), "GVL(전역변수리스트)"이 표시될 수 있습니다.
4. "GVL(전역변수리스트) 변수"(GVL variables) 아래에서 "A1\_yellow" 변수를 선택합니다.
5. 맨 아래쪽에 있는 신호등을 강조합니다.
6. "GVL(전역변수리스트) 변수"(GVL variables) 아래에서 "A1\_green" 변수를 선택합니다.

## 첫 단계 - IndraLogic

**"신호등 하우징" 요소 추가 및 편집**

**신호등 하우징의 요소** 이제 신호 상태를 표시하기 위한 세 개의 신호등이 완성되었습니다. 신호등(평선블럭)의 모양을 실제 신호등과 유사하게 조정해야 합니다. 신호등의 시각적 표시에 다음 세 가지 요소를 추가합니다.

1. 신호등 하우징
2. 신호등 하우징의 기둥
3. 신호등베이스

**"직사각형" 요소 추가** 신호등 하우징을 추가하려면 다음 단계를 진행합니다.

1. "도구 상자"(ToolBox) 창에서 "직사각형"(Rectangle) 요소를 선택합니다.
2. 시각화 편집기에서 비어 있는 영역을 클릭하여 "직사각형"(Rectangle) 요소를 삽입합니다.

**"신호등 하우징" 요소의 위치 변경** 덮개(평선블럭)의 모양과 위치를 변경하려면 다음 단계를 진행합니다.

1. 추가한 "직사각형"(Rectangle) 요소를 강조하고 "속성"(Properties) 창을 엽니다.
2. "속성"(Properties) 창의 "위치"(Position)에서 (X, Y)에 대해 (50, 135)를 선택하여 위치를 변경합니다.
3. "속성"(Properties) 창의 "위치"(Position) 아래에서 (너비, 높이)에 대해 (50, 150)을 선택하여 모양을 변경합니다.
4. 중심점을 변경하려면 "중심"(Center) 아래에서 (X, Y)에 대해 (50, 210)을 선택합니다.
5. 강조된 개체 옆을 마우스 오른쪽 버튼으로 클릭하여 상황에 맞는 메뉴를 엽니다. **시퀀스(Sequence) ▶ 맨 뒤로 보내기(Send to back)**를 선택합니다.

**"신호등 하우징" 요소의 속성 변경** 신호등 하우징의 채우기 색을 변경하려면 다음 단계를 진행합니다.

1. "속성"(Properties) 창의 "색"(Colors), "정상 상태"(Normal state), "채우기 색"(Fill color) 아래에서 "..." 버튼을 클릭합니다.  
"색"(Color) 창이 열립니다.
2. "색"(Color) 창에서 "색 정의"(Define Colors)를 클릭합니다.
3. "빨간색"(Red), "녹색"(Green) 및 "파란색"(Blue) 필드에 다음과 같이 입력합니다.  
(빨간색, 녹색, 파란색) = (5, 5, 10)

**"신호등 기둥" 요소 추가 및 편집**

**"직사각형" 요소 추가** 다음으로는 신호등 기둥을 만듭니다.  
다음과 같이 진행합니다.

1. "도구 상자"(ToolBox) 창에서 "직사각형"(Rectangle) 요소를 선택합니다.
2. 시각화 편집기에서 비어 있는 영역을 클릭하여 "직사각형"(Rectangle) 요소를 삽입합니다.

- "신호등 기둥" 요소의 위치 변경**
1. 추가한 "직사각형"(Rectangle) 요소를 강조하고 "속성"(Properties) 창을 엽니다.
  2. "속성"(Properties) 창의 "위치"(Position)에서 (X, Y)에 대해 (70, 285)를 선택하여 위치를 변경합니다.
  3. "속성"(Properties) 창의 "위치"(Position) 아래에서 (너비, 높이)에 대해 (10, 195)을 선택하여 모양을 변경합니다.
  4. 중심점을 변경하려면 "중심"(Center) 아래에서 (X, Y)에 대해 (75, 380)을 선택합니다.

## 첫 단계 - IndraLogic

### "신호등 기둥" 요소의 속성 변경

신호 램프 기둥의 채우기 색으로 진한 회색(DimGray)(색, 정상 상태, 채우기 색, (빨간색, 녹색, 파란색) = (105, 105, 105))을 선택합니다.

### "신호등베이스" 요소 추가 및 편집

#### "직사각형" 요소 추가

첫 번째 신호등에는 베이스가 필요합니다.

신호등베이스를 만들려면 다음 단계를 진행합니다.

1. "도구 상자"(ToolBox) 창에서 "직사각형"(Rectangle) 요소를 선택한 후 작업 영역에 추가합니다.
2. 추가한 "직사각형"(Rectangle) 요소를 강조하고 "속성"(Properties) 창을 엽니다.
3. "속성"(Properties) 창의 "위치"(Position)에서 (X, Y)에 대해 (15, 480)를 선택하여 위치를 변경합니다.
4. "속성"(Properties) 창의 "위치"(Position) 아래에서 (너비, 높이)에 대해 (120, 30)을 선택하여 모양을 변경합니다.
5. 중심점을 변경하려면 "중심"(Center) 아래에서 (X, Y)에 대해 (75, 495)을 선택합니다.

#### "신호등베이스" 요소의 속성 변경

신호 램프 기둥의 채우기 색으로 진한 회색(DimGray)(색, 정상 상태, 채우기 색, (빨간색, 녹색, 파란색) = (105, 105, 105))을 선택합니다.

첫 번째 신호등(평선블럭)의 모양은 다음과 같습니다.

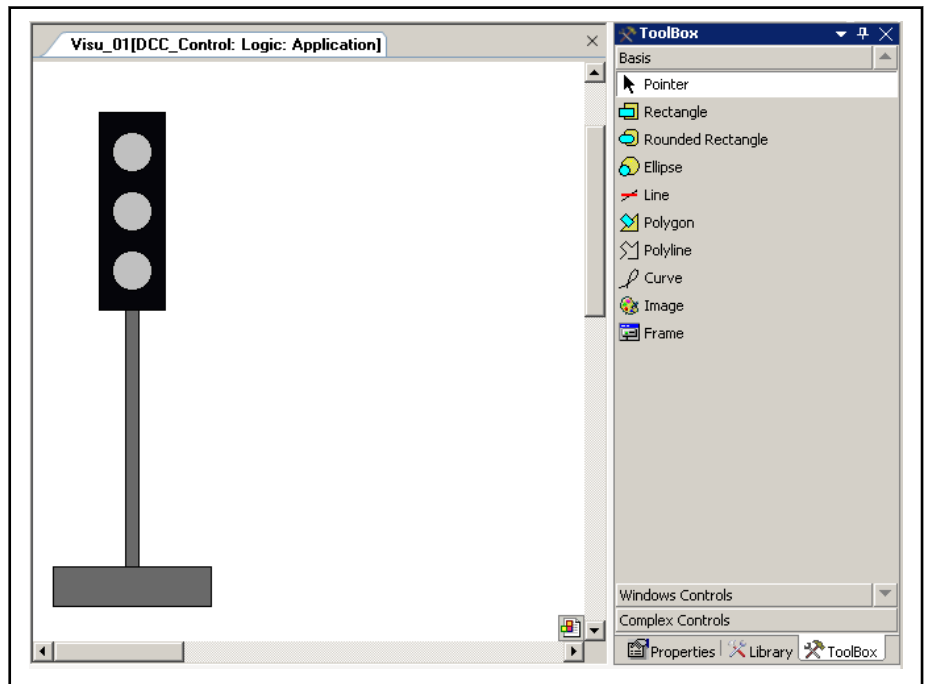


그림 4-72: 첫 번째 신호등

### 신호등 두 개 추가 및 편집

#### 두 번째 신호등

두 번째 신호등을 만들려면 모든 신호등 요소를 강조합니다. 신호등의 오른쪽 아래 영역을 마우스 왼쪽 버튼으로 클릭하고 마우스 버튼을 누르고 있습니다. 마우스 포인터를 신호등의 왼쪽 위로 끈 다음 마우스 버튼을 놓습니다. 모든 요소가 강조됩니다.



또는 <Ctrl> 키를 누른 상태에서 원하는 요소를 강조하여 개별 요소를 강조할 수도 있습니다.

상황에 맞는 메뉴에서 복사(Copy) 및 삽입(Insert)을 선택합니다. 첫 번째 신호등 옆에 강조된 개체를 저장합니다.

## 첫 단계 - IndraLogic

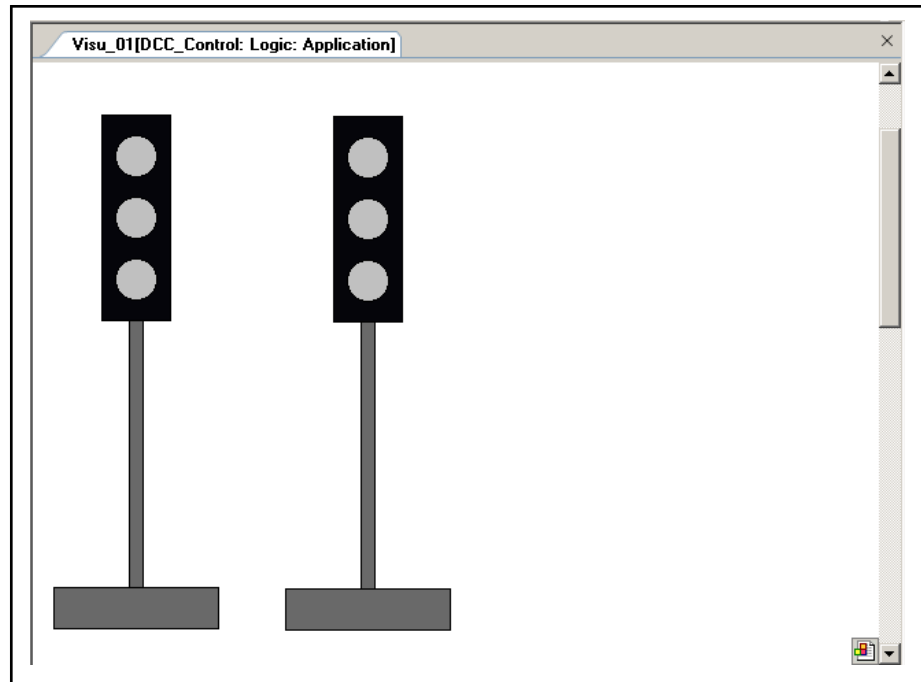


그림 4-73: 신호등 두 개

다양한 신호 단계 간에 전환하기 위한 변수를 조정해야 합니다. "속성"(Properties) 창의 "색 전환"(ToggleColor)에서 변수를 적절히 수정합니다.

"GVL.A1\_red"는 "GVL.A2\_red"가 됩니다.

"GVL.A1\_yellow"는 "GVL.A2\_yellow"가 됩니다.

마지막으로 "GVL.A1\_green"은 "GVL.A2\_green"이 됩니다.

## ON 스위치 추가 및 편집

**"버튼" 요소 추가**

여기에서는 신호등을 시작 단계로 전환할 수 있습니다. 다음과 같이 "버튼"(button)을 삽입해야 합니다.

1. **도구 상자(ToolBox) ▶ 창 컨트롤(Windows Control)**에서 "버튼"(button) 요소를 선택합니다.
2. 시각화 편집기에서 비어 있는 영역을 클릭하여 "버튼"(button) 요소를 삽입합니다.

**"버튼" 요소의 위치 변경**

1. "속성"(Properties) 창의 "위치"(Position)에서 (X, Y)에 대해 (130, 55)를 선택하여 위치를 변경합니다.
2. "속성"(Properties) 창의 "위치"(Position) 아래에서 (너비, 높이)에 대해 (70, 40)을 선택하여 모양을 변경합니다.
3. 중심점을 변경하려면 "중심"(Center) 아래에서 (X, Y)에 대해 (165, 75)을 선택합니다.

**"버튼" 요소의 속성 변경**

1. "텍스트"(Texts) 속성의 "텍스트"(Text) 아래에 "ON"을 입력합니다.
2. "입력"(Input) 속성의 "OnClick" 아래에서 "구성"(Configure)을 클릭합니다.  
"입력 구성"(Input Configuration) 창이 열립니다.
3. "입력 구성"(Input Configuration) 창의 왼쪽 창 영역에서 "스위치 변수"(Switch variable)를 강조하고 ">"을 클릭합니다.  
"스위치 변수"(Switch variable)가 오른쪽 창 영역에 추가됩니다.
4. "변수"(Variable) 입력 필드를 클릭하고 "GVL.ON"을 입력합니다.

5. "확인"(OK) 버튼을 사용하여 입력을 확인합니다.

## 레이블 추가 및 편집

- "직사각형" 요소 추가** 이제 신호등 두 개에 "신호등 1" 및 "신호등 2"로 레이블을 지정해야 합니다. 신호등의 레이블을 지정하려면 다음 단계를 진행합니다.
1. "도구 상자"(ToolBox) 창에서 "직사각형"(Rectangle) 요소를 선택합니다.
  2. 시각화 편집기에서 비어 있는 영역을 클릭하여 "직사각형"(Rectangle) 요소를 삽입합니다.
- "직사각형" 요소의 위치 변경**
1. 추가한 "직사각형"(Rectangle) 요소를 강조하고 "속성"(Properties) 창을 엽니다.
  2. "속성"(Properties) 창의 "위치"(Position)에서 (X, Y)에 대해 (15, 520)를 선택하여 위치를 변경합니다.
  3. "속성"(Properties) 창의 "위치"(Position) 아래에서 (너비, 높이)에 대해 (120, 30)을 선택하여 모양을 변경합니다.
  4. 중심점을 변경하려면 "중심"(Center) 아래에서 (X, Y)에 대해 (75, 535)을 선택합니다.
- "직사각형" 요소의 속성 변경**
1. "텍스트"(Text) 옆의 "텍스트"(Texts) 필드에 "신호등 1"(Traffic light 1)을 입력합니다.
  2. "속성"(Properties) 창의 "색"(Colors), "정상 상태"(Normal state), "프레임 색"(Frame color) 아래에서 "..." 버튼을 클릭합니다.  
"색"(Color) 창이 열립니다.
  3. "색"((Color)) 창에서 "색 정의"(Define color) 버튼을 클릭합니다.
  4. "빨간색"(Red), "녹색"(Green) 및 "파란색"(Blue) 필드에 다음과 같이 입력합니다. (빨간색, 녹색, 파란색)= (255, 255, 255)
- "직사각형" 요소 복사 및 조정**
1. 레이블을 복사한 후 두 번째 신호등 아래에 배치합니다.
  2. "텍스트"(Text) 옆의 "텍스트"(Texts) 필드에 "신호등 2"(Traffic light 2)을 입력합니다.
- 다음과 같이 시각화가 완료되었습니다.

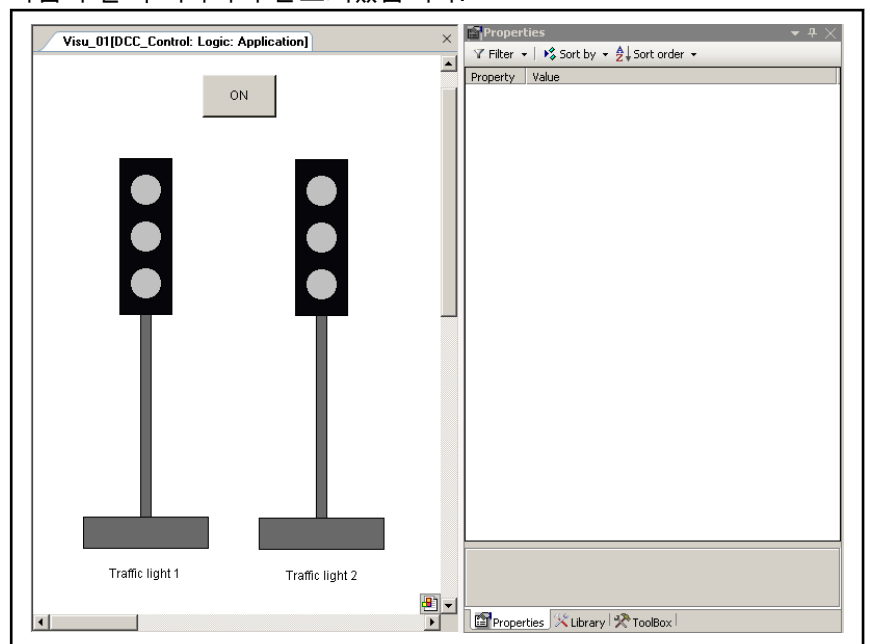


그림 4-74: 시각화 편집기의 완료된 시각화

## 첫 단계 - IndraLogic

## 변수 값 표시

변수의 값을 시각화를 가동하기 위해 형식이 지정된 방식으로 표시하거나 영구 솔루션으로 표시하면 유용할 수 있습니다.

**"직사각형" 요소 추가**

두 신호등의 "신호등 1"(Traffic light 1) 및 "신호등 2"(Traffic light 2) 레이블 아래에 각각 "STATUS" 변수가 표시됩니다.

신호등의 레이블을 지정하려면 다음 단계를 진행합니다.

1. "도구 상자"(ToolBox) 창에서 "직사각형"(Rectangle) 요소를 선택합니다.
2. 시각화 편집기에서 비어 있는 영역을 클릭하여 "직사각형"(Rectangle) 요소를 삽입합니다.

**"직사각형" 요소의 위치 변경**

1. 추가한 "직사각형"(Rectangle) 요소를 강조하고 "속성"(Properties) 창을 엽니다.
2. "속성"(Properties) 창의 "위치"(Position)에서 (X, Y)에 대해 (15, 570)를 선택하여 위치를 변경합니다.
3. "속성"(Properties) 창의 "위치"(Position) 아래에서 (너비, 높이)에 대해 (120, 30)을 선택하여 모양을 변경합니다.
4. 중심점을 변경하려면 "중심"(Center) 아래에서 (X, Y)에 대해 (75, 585)을 선택합니다.

**"직사각형" 요소의 속성 변경**

1. "텍스트"(Text) 아래의 "텍스트"(Texts) 필드에 "신호등 1: %s"(Traffic light 1: %s)를 입력합니다.
2. "텍스트 변수"(Text Variable) 필드에서 "텍스트 변수"(Text variables)를 클릭한 다음 "..."을 클릭하여 "PlcProg.LIGHT1.STATUS" 변수를 선택합니다.

온라인 모드의 형식 지정 "%s" 대신 "PlcProg.LIGHT1.STATUS" 변수의 값이 나타납니다.

**"직사각형" 요소 복사 및 조정**

1. 화면 표시를 복사한 후 두 번째 신호등 아래에 배치합니다.
  2. 화면 표시 2에 대해 다음과 같이 바꿉니다.
    - 텍스트 "신호등 1: %s"(Traffic light 1: %s)를 "신호등 2: %s"(Traffic light 2: %s)로 바꿉니다.
    - 텍스트 변수 "PlcProg.LIGHT1.STATUS"를 "PlcProg.LIGHT2.STATUS"로 바꿉니다.
- 화면 표시가 완료된 시각화는 다음과 같습니다.

## 첫 단계 - IndraLogic

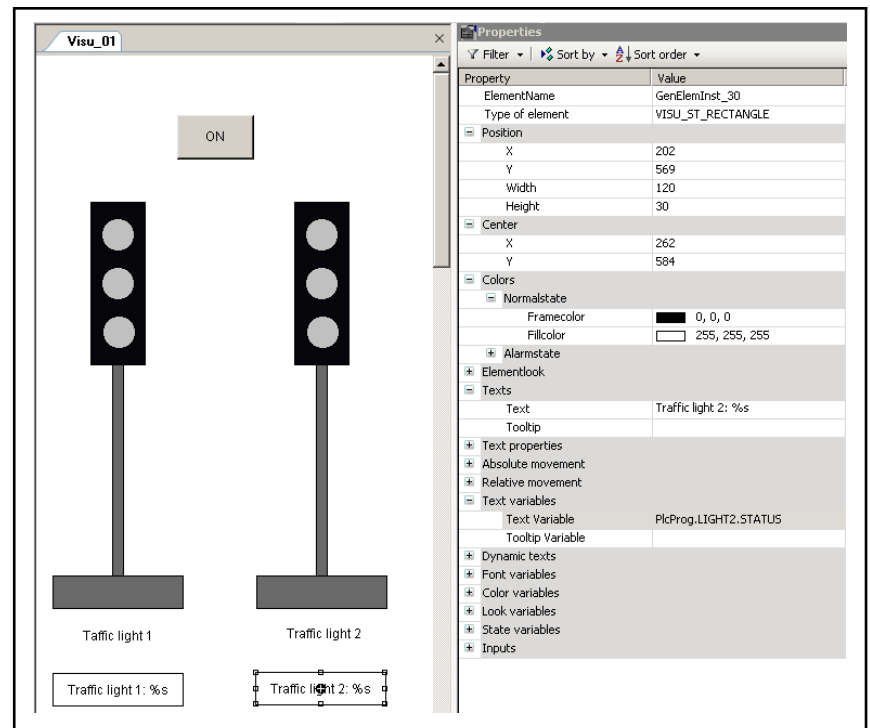


그림 4-75: 화면 표시가 완료된 시각화 편집기

### 4.7.4 시각화 테스트

시뮬레이션 모드에서 시각화를 테스트할 수 있습니다. 실제 컨트롤을 사용할 수 있는 경우에는 실제 컨트롤에서 테스트할 수 있습니다.

시뮬레이션 모드에서 시각화를 테스트하려면 "시뮬레이션 모드에서 응용 프로그램 테스트(ST(구조체언어)로 작성한 PlcProg)"(58 페이지) 또는 "시뮬레이션 모드에서 응용 프로그램 테스트(CFC(연결구조언어)로 작성한 PlcProg)"(63 페이지)를 참조하십시오.

컨트롤을 사용하여 시각화를 테스트하려면 "컨트롤에서 응용 프로그램 테스트"(65 페이지)를 참조하십시오.

응용 프로그램이 시작되면 "시각화"(Visualization) 창을 엽니다.

## 첫 단계 - IndraLogic

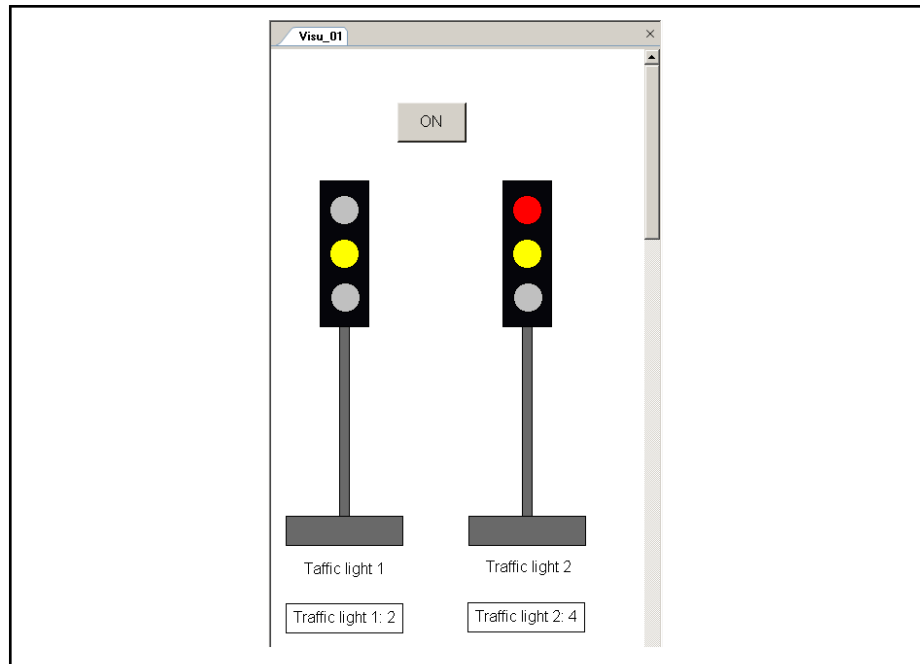


그림 4-76: 온라인 시각화

"ON"을 클릭합니다. 신호등 주기가 처리됩니다.

## 4.8 프로젝트 복원

**위치** 만든 프로젝트를 기반으로 하여, 제어 하드웨어(XLC L25, L45, L65(sercos III 버스 포함 또는 제외) 또는 MLC L25, L45, L65)에 관계없이 프로젝트를 다시 생성할 수 있는 내보내기 파일이 생성되었습니다. 다시 생성된 프로젝트는 시뮬레이션으로 실행하거나 컨트롤을 사용하여 실행할 수 있습니다.

내보내기 파일은 "응용 프로그램"(Application) 노드부터 생성되었으며 다음과 같은 구성 요소를 포함합니다.

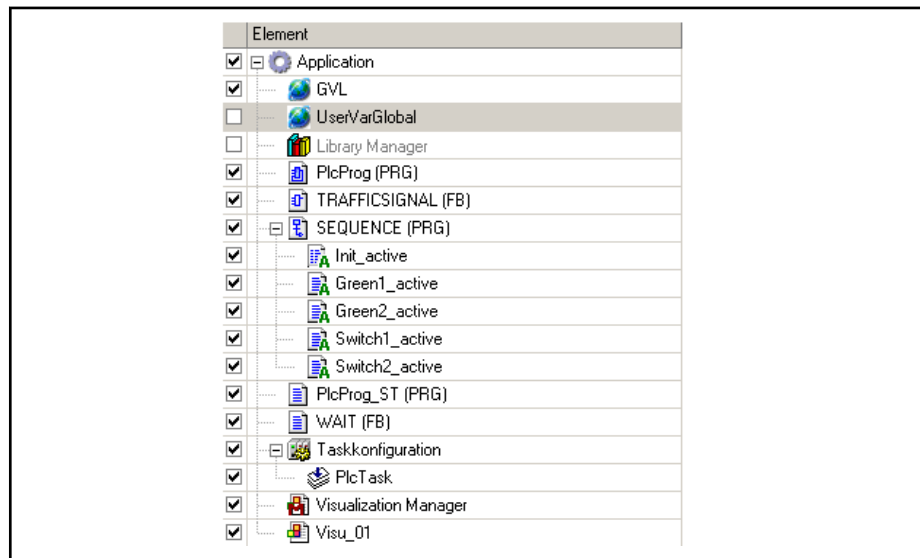


그림 4-77: 내보내기 파일의 구성 요소

가져오는 동안 제공된 구성 요소를 덮어씁니다.

첫 단계 - IndraLogic

- 복원**    선택한 컨트롤(XLC L25, L45, L65(sercos III 버스 포함 또는 제외) 또는 MLC L25, L45, L65)에서 프로젝트가 실행되도록 하려면 약간의 사용자 개입이 필요합니다.
1.    컨트롤에서 이전 데이터 지우기  
컨트롤의 이전 메모리 내용을 지웁니다.  
자세한 내용은 [컨트롤 재설정\(CMLx5\)\(24 페이지\)](#)에 설명되어 있습니다.  
컨트롤의 메모리가 지워지고 가능한 모션 구성 요소가 나타나고 PLC 구성 요소가 작동할 준비가 됩니다.
  2.    새 프로젝트 만들기  
빈 프로젝트를 새로 만듭니다. **파일(File) ▶ 새로 만들기(New) ▶ 프로젝트(Project)**. "첫 단계"(First Steps)가 권장되기는 하지만 프로젝트 이름은 임의로 선택할 수 있습니다.  
빈 프로젝트가 새로 생성됩니다.
  3.    컨트롤 만들기  
선택한 컨트롤(XLC L25, L45, L65(sercos III 버스 포함 또는 제외) 또는 MLC L25, L45, L65)을 제품라이브러리(Device Library)에서 빈 프로젝트로 끌어다 놓습니다. "DCC\_Control"이 권장되기는 하지만 컨트롤 이름은 임의로 선택할 수 있습니다.  
제품유형 및 IP 주소를 지정합니다.  
연결 테스트를 수행합니다. "시뮬레이션"(Simulation) 모드에서만 작동하는 경우에는 필요하지 않습니다.  
평소와 같이 추가를 끝냅니다.  
이제 프로젝트에 원하는 컨트롤이 있습니다.
  4.    내보내기 파일 "Application\_FirstSteps\_IndraLogic\_Basics.iwx" 가져오기  
프로젝트 탐색기에서 컨트롤 노드를 **로직(Logic) ▶ 응용 프로그램(Application)**까지 확장합니다.  
상황에 맞는 메뉴: "응용 프로그램(Application)" **가져오기(Import)**.  
"Iw:\All Programs\Rexroth\IndraWorks\Projects"에서 내보내기 파일 "Application\_FirstSteps\_IndraLogic\_Basics.iwx"를 선택하고 "덮어쓰기"(Overwrite) 작업을 수행한 후 완료(Finish)를 클릭합니다.  
필요한 파일이 복사되었고, 프로젝트가 시뮬레이션 모드가므로 프로젝트를 컨트롤에서 실행할 수 있습니다.

**프로젝트 테스트**



프로젝트에는 CFC(연결구조언어)로 작성한 "PlcProg" 및 ST(구조체언어)로 작성한 "PlcProg\_ST"가 포함되어 있습니다.

시뮬레이션(Simulation) 모드에서 프로젝트를 테스트하려면 ["시뮬레이션 모드에서 응용 프로그램 테스트\(CFC\(연결구조언어\)로 작성한 PlcProg\)"\(58 페이지\)](#) 또는 ["시뮬레이션 모드에서 응용 프로그램 테스트\(ST\(구조체언어\)로 작성한 PlcProg\)"\(58 페이지\)](#)를 참조하십시오.

컨트롤을 사용하여 프로젝트를 테스트하려면 ["컨트롤에서 응용 프로그램 테스트"\(65 페이지\)](#)를 참조하십시오.



## 5 첫 단계 - 모션

### 5.1 프로젝트 목표 "첫 단계 - 모션"

일정 속도로 실행하도록 되어 있는 가상 축이 지정되어 있습니다. 이 축은 기어를 통해 커플링된 실제 축의 마스터 축 역할을 합니다.

예를 들어, 실제 축에는 sercos III 케이블을 통해 컨트롤에 연결된 "IndraDrive M" 드라이브가 포함되어 있습니다.



드라이브가 없는 경우 다음 단계 "드라이브 스캔"이 생략됩니다.  
실제 축 대신 두 번째 가상 축을 만들고 이 가상 축을 실제 축 대신 사용하십시오.

#### 하위 작업

- IndraWorks 프로젝트, 컨트롤 및 드라이브 만들기
- 컨트롤에서 모션 프로젝트 로드
- 모션 스킨스 프로그래밍
- PLC 프로그램을 컨트롤에 로드한 후 테스트
- 트레이스를 사용하여 축 모션 기록



모션 스킨스는 FBD(평선블럭다이어그램) 또는 ST(구조체언어)로 프로그래밍할 수 있습니다.

- [FBD\(평선블럭다이어그램\)에서 모션 스킨스 프로그래밍, 91 페이지](#)
- [ST\(구조체언어\)에서 모션 스킨스 프로그래밍, 115 페이지](#)

### 5.2 기술 요구 사항

- IndraMotion MLC 용 IndraWorks Engineering 이 설치된 PC, 12VRS, 컨트롤, 유형 "IndraMotion MLC L25", "IndraMotion MLC L45" 또는 "IndraMotion MLC L65",  
최신 펌웨어가 로드되어 있어야 합니다.

이러한 사전 요구 사항 충족에 대해서는 다음 설명서를 참조하십시오.

- Rexroth IndraMotion MLC 12VRS 기능 설명,  
DOK-IM\*ML\*-SYSTEM\*\*V12-RExx-EN-P 및
- IndraWorks 온라인 도움말, **System IndraLogic XLC/IndraMotion MLC ▶ XLC/MLC 기능 설명(XLC/MLC Functional Description) ▶ 가동 및 작동(Commissioning and Operation)**,

컨트롤(예: 유형 "IndraMotion MLC L65.1"), 최신 펌웨어가 로드되어 있어야 합니다.

- IP 주소를 알아야 합니다.
- 컨트롤이 이더넷을 통해 프로그래밍 장치(PC)에 연결되어 있어야 합니다.

드라이브 컨트롤러(예: 유형 "IndraDrive M"), 최신 펌웨어가 로드되어 있어야 합니다.

- 드라이브 컨트롤러가 sercos III 을 통해 컨트롤에 연결되어 있어야 합니다.
- sercos 버스에서 드라이브 컨트롤러의 주소를 스캔해야 합니다.

또는

- 드라이브가 없는 경우 브리지로 연결된 sercos 버스를 스캔해야 합니다.

## 첫 단계 - 모션

## 5.3 모션 하위 프로젝트 만들기

## 5.3.1 컨트롤 재설정(CMLx5)

컨트롤에는 이전 프로그램 구성 요소, 매개변수 입력 및 PLC 데이터 등 이전 사용 기록이 남아 있을 수 있습니다. 이 단계에서는 메모리를 비워 이러한 데이터를 제거합니다.



그림 5-1: 컨트롤 다시 부팅

컨트롤의 전원이 켜진 후에 디스플레이 아래에 있는 외부 버튼 <ESC>와 <Enter>를 누르고 부팅 1.02(BOOT 1.02)가 나타난 다음 부팅 중지(BOOTSTOP)가 나타날 때까지 버튼을 누르고 있습니다. 두 개의 내부 버튼을 누릅니다. 먼저 정리(CLEANUP)를 누른 다음 다시 부팅(REBOOT)을 누릅니다.

이제 BB 중지(BB STOP)가 표시될 때까지 컨트롤이 실행되고, 메모리가 비워지며, 모션 구성 요소가 작동할 준비가 되고, 처리 중인 PLC 구성 요소가 중지됩니다.

## 5.3.2 IndraWorks 시작

시작(Start) ▶ 프로그램(Programs) ▶ Rexroth ▶ IndraWorks ▶ Engineering 을 통해 IndraWorks 를 시작합니다.

IndraWorks Engineering 환경이 시작됩니다. 또는 바탕 화면 아이콘을 통해 IndraWorks 를 시작할 수도 있습니다.



바탕 화면 아이콘: 

## 5.3.3 새 프로젝트 만들기

새 프로젝트를 만들려면 작업 공간에서 "빈 프로젝트 만들기"(Create an empty project)를 클릭합니다.

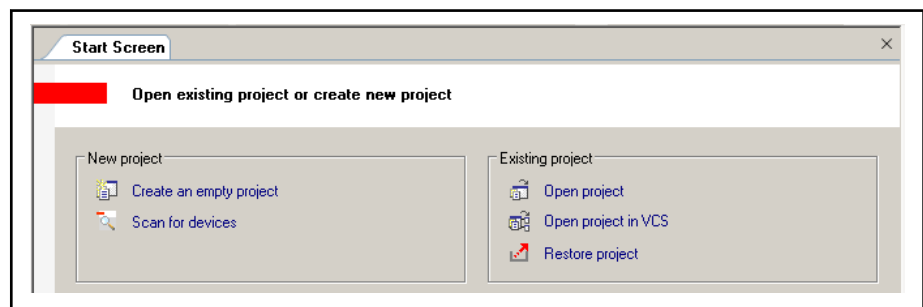


그림 5-2: 빈 프로젝트 만들기

또는 파일(File) ▶ 새로 만들기(New) ▶ 프로젝트...(Project...)를 통해 새 프로젝트를 만들 수도 있습니다.

새 IndraWorks 프로젝트의 프로젝트 이름 및 저장 위치 설정을 지정할 수 있는 마법사가 시작됩니다. 다음 그림을 참조하십시오.

첫 단계 - 모션

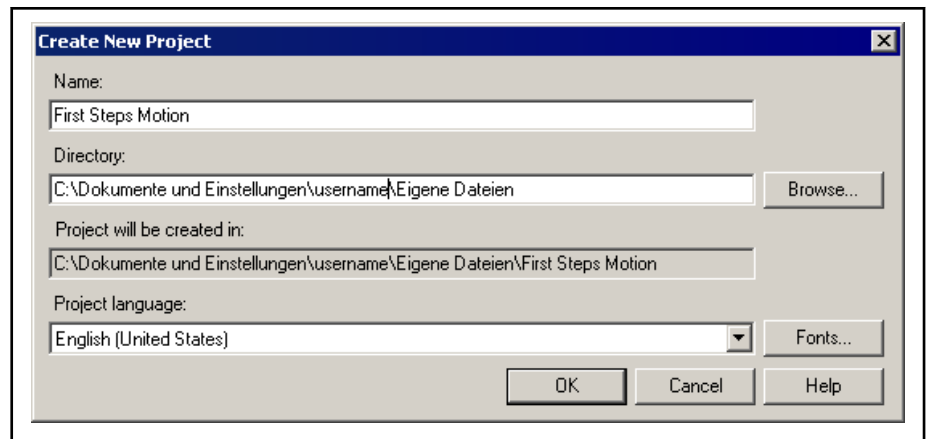


그림 5-3: 새 IndraWorks 프로젝트를 저장하는 설정

- 이름(Name)** 프로젝트의 이름을 지정합니다. 제안: "첫 단계 - 모션"(First Steps - Motion)
- 디렉터리(Directory)** 이 필드에는 기본적으로 C:\Documents and Settings\username\MyDocuments 디렉터리의 저장 위치가 포함되어 있습니다.  
다른 디렉터리를 지정할 수도 있습니다.
- 프로젝트를 만들 위치(Project is created in)** 디렉터리 및 프로젝트 이름이 자동으로 생성됩니다.
- 프로젝트 언어(Project language)** 프로젝트 언어를 선택하면 사용자 텍스트의 언어가 결정됩니다.  
프로젝트 탐색기에 이전에 정의한 프로젝트 이름을 사용하여 새로 만든 프로젝트가 나타납니다. 다음 그림을 참조하십시오.

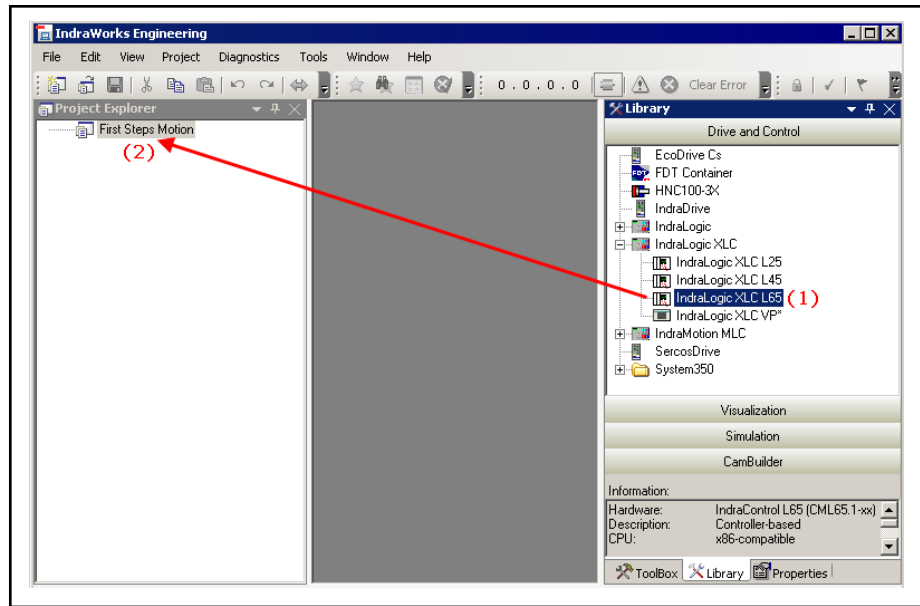


그림 5-4: 첫 단계 - 모션 프로젝트

### 5.3.4 프로젝트에 새 제품 추가

"드라이브 및 컨트롤"(Drive and Control) 라이브러리에서 원하는 제품(예: "IndraLogic XLC L65")를 프로젝트 탐색기의 새로 만든 프로젝트로 끌어다 놓습니다.

## 첫 단계 - 모션



- (1) 제품: IndraLogic XLC L65  
(2) 프로젝트: 첫 단계 - 모션(First Steps - Motion)

그림 5-5: 제품 추가

그러면 아래 설명된 대로 필요한 기본 설정을 지정할 수 있는 마법사가 자동으로 시작됩니다.

## 5.3.5 제품에 대한 기본 설정 지정

## 제품설정

추가할 제품에 대한 일반적인 정보를 입력합니다.

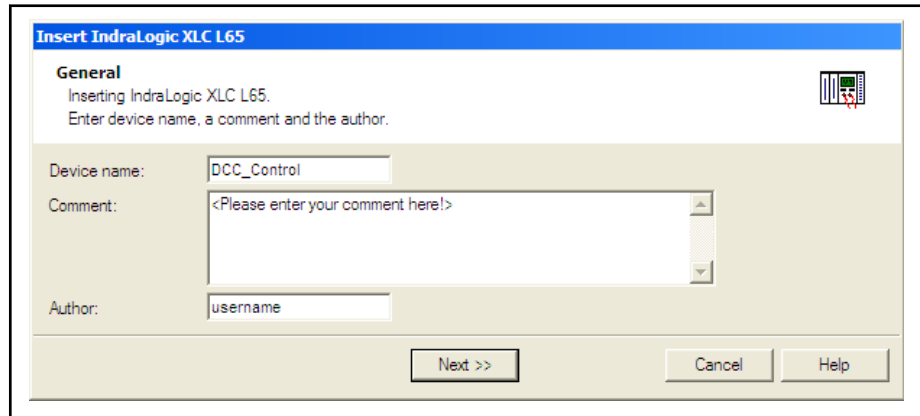


그림 5-6: 제품설정

- 제품 이름(Device name)** "제품 이름"(Device Name)에는 원하는 제품 이름(예: "DCC\_Control")을 입력합니다. 기본적으로 이 필드에는 라이브러리에 표시된 이름이 포함되어 있습니다. 나중에 제품은 프로젝트 탐색기에 지정된 이름으로 나타납니다.
- 설명(Comment)** 제품에 대한 자세한 설명을 추가합니다. 이 필드는 선택 사항입니다.
- 작성자(Author)** 프로젝트 작성자의 이름을 입력합니다. 이 필드는 선택 사항입니다.



이러한 입력 사항은 나중에 변경할 수도 있습니다.

"다음>>"(Next>>)을 클릭하여 입력 사항을 확인합니다.

## 통신 설정

"구성" 대화 상자에서 펌웨어 및 IndraWorks 와 제품(이 경우 IndraLogic XLC L65) 간 통신에 대한 설정을 정의합니다.

그림 5-7: 구성

|                          |                                                                                                                                                                                                                                                                                                                                                                                            |
|--------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 제품유형(Device type)        | 제품유형은 컨트롤 선택을 통해 미리 결정됩니다. 하나의 제품유형에 대해 여러 가지 다른 구성이 있는 경우 수동 조정이 필요합니다. 예를 들어, IndraLogic XLC Lx5 는 sercos III 버스와 함께 사용하거나(CMLx5.1-3P), sercos III 버스 없이 사용할 수 있습니다(CMLx5.1-NP).                                                                                                                                                                                                         |
| 펌웨어 버전(Firmware version) | 여기에서는 펌웨어 버전을 선택합니다. 일반적으로 이 설정은 유지될 수 있습니다.                                                                                                                                                                                                                                                                                                                                               |
| 펌웨어 출시(Firmware release) | 여기에서는 펌웨어 버전 내의 펌웨어 출시를 설정합니다. 이 설정도 유지되어야 합니다.                                                                                                                                                                                                                                                                                                                                            |
| IP 주소(IP address)        | "제품 IP 주소"에 대상 주소(즉, 응용 프로그램을 로드해야 하는 컨트롤의 IP 주소)를 입력합니다.                                                                                                                                                                                                                                                                                                                                  |
| PLC 게이트웨이(PLC Gateway)   | "게이트웨이 IP 주소"에서 게이트웨이 서버가 실행 중인 컴퓨터의 주소를 정의합니다. 여기에서 기본 컴퓨터 이름(주소)은 "로컬 호스트"(localhost)입니다. 즉, 로컬로 설치된 게이트웨이의 주소를 지정합니다. 이 설정은 변경되지 않습니다.                                                                                                                                                                                                                                                  |
| 연결 테스트(Connection test)  | <p>"연결 테스트"(Connection Test) 버튼을 클릭하여 컴퓨터와 대상 컨트롤 간 통신이 성공적으로 수행되는지 확인합니다. "연결 테스트 결과"(Connection test result) 필드에 테스트 결과가 기록됩니다.</p> <p>PLC 통신 테스트가 성공적으로 수행된 경우 IP 주소가 표시됩니다.</p> <p>발생할 가능성이 있는 모든 문제는 "연결 테스트 결과"(Connection Test Result) 창에 빨간색 또는 노란색의 깜박이는 아이콘으로 표시됩니다.</p> <p>다음과 같은 문제가 발생할 수 있습니다.</p> <ul style="list-style-type: none"> <li>출력: "컨트롤과의 통신에 실패했습니다."</li> </ul> |

## 첫 단계 - 모션

원인:

- 컨트롤의 전원이 켜져 있지 않습니다.
- 네트워크 케이블이 연결되어 있지 않습니다.
- 컨트롤의 IP 주소와 설정된 IP 주소가 일치하지 않습니다.

해결 방안: 대화 상자에서 IP 주소를 변경하거나 컨트롤의 IP 주소를 변경합니다. Rexroth IndraMotion MLC 기능 설명의 컨트롤 가동을 참조하십시오.

- PC 및/또는 컨트롤의 네트워크 마스크가 너무 작게 설정되었습니다.

해결 방안: 네트워크 설정을 확인하고 수정합니다. Rexroth IndraMotion MLC 기능 설명의 컨트롤 가동을 참조하십시오.

- 출력: "구성된 펌웨어와 컨트롤의 펌웨어가 일치하지 않습니다."

원인:

- 컨트롤 펌웨어가 최신 버전이 아닙니다.

해결 방안: 펌웨어를 업데이트합니다. Rexroth IndraMotion MLC 기능 설명의 컨트롤 가동을 참조하십시오.

- 선택한 컨트롤이 실제 컨트롤 유형과 일치하지 않습니다.

안전한 온라인 모드(Secure online mode)

이 옵션을 선택하면 보안 관련 작업을 실행하기 전에 확인하는 프롬프트가 나타납니다.

PLC 프로그래밍 언어(PLC programming language)

여기에서 기본 PLC 프로그래밍 언어를 설정합니다. ST(구조체언어)를 선택합니다.

"다음 >>"(Next >>)을 클릭하여 다음 페이지로 이동합니다.

## 평선모듈설정



제품유형에 따라 "제품 및 평선모듈설정" 대화 상자가 달라질 수 있습니다. 이 프로젝트에서는 "IndraLogic XLC L65" 제품의 대화 상자를 예로 들어 설명합니다.

원하는 제품 및 평선모듈설정을 입력합니다.

**Insert IndraLogic XLC L65**

**Interfaces**  
Please select which onboard components and function modules you want to use.

SERCOSIII configuration (X7E1/X7E2): SercosIII Master

Slot0 Configuration (X7P): Profibus DP Master

Slot1 Configuration (X7E3/X7E4): Nicht verwendet

Ethernet configuration (X7E5): Not used

Function modules: Not used

Not used

Not used

Not used

Not used

그림 5-8: 평선모듈설정

sercos III 구성(sercos III configuration)

제품을 추가할 때 선택할 수 있는 현재 sercos III 구성입니다. "sercos III 마스터"(sercos III Master) 또는 "사용하지 않음"(Not used) 중에서 선택할 수 있습니다.

첫 단계 - 모션

- 슬롯 0 구성(Slot0 configuration)** 제품을 추가할 때 선택할 수 있는 현재 PROFIBUS 구성입니다. "PROFIBUS DP 마스터"(PROFIBUS DP Master), "PROFIBUS DP 슬레이브"(PROFIBUS DP Slave) 또는 "사용하지 않음"(Not Used) 중에서 선택할 수 있습니다.
- 슬롯 1 구성(Slot1 configuration)** 제품을 추가할 때 선택할 수 있는 현재 PROFINET 또는 이더넷/IP 구성입니다. "사용하지 않음"(Not Used), PROFINET 또는 이더넷/IP 인터페이스 중에서 선택할 수 있습니다.
- 이더넷 구성(EtherNet configuration)** 이 인터페이스는 Engineering 인터페이스로 필요하므로(PC에 직접 연결 또는 작동 네트워크를 통해 연결) 이더넷/IP 어댑터가 사용되지 않을 수 있습니다.
- 평선모듈(Function Modules) 1-4** 평선모듈은 최대 4 개까지 "IndraLogic XLC L65"에 연결할 수 있습니다. 필요한 경우 각 목록에서 선택할 수 있습니다. "완료"(Finish)를 클릭하여 입력 사항을 확인합니다. 프로젝트 탐색기에 새로 만든 제품이 나타납니다.

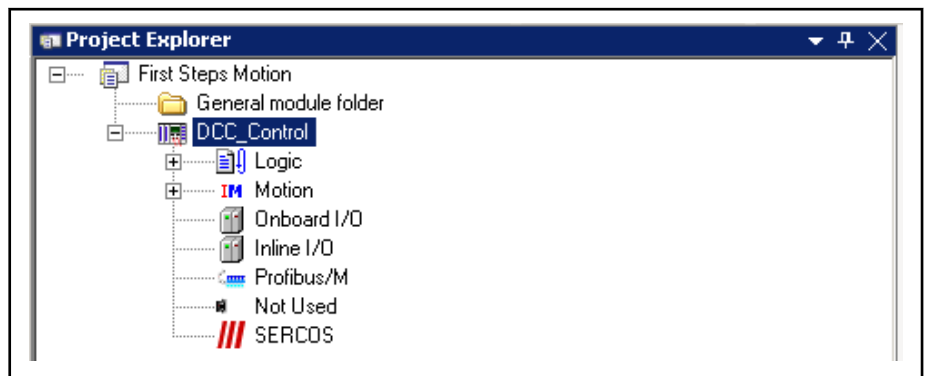


그림 5-9: 프로젝트 탐색기: sercos III 버스를 사용하는 IndraLogic XLC L65 컨트롤이 연결된 "첫 단계 - 모션"(First Steps - Motion) 프로젝트



프로젝트 탐색기의 화면에는 "구성"(Configuration) 및 "평선모듈"(Function Modules) 대화 상자에서 선택한 구성 요소가 반영됩니다.

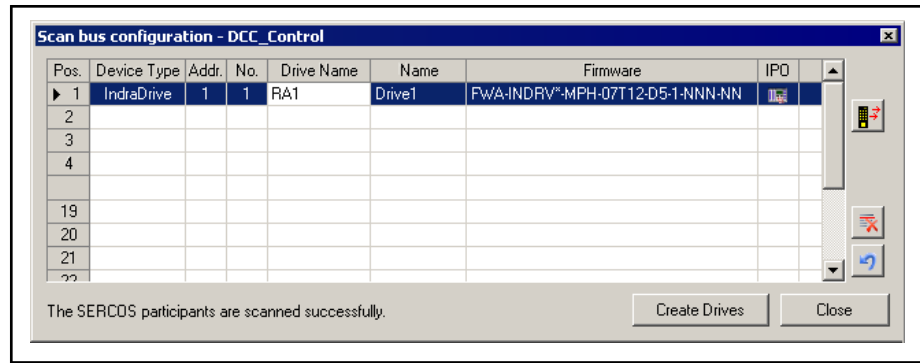
### 5.3.6 프로젝트에 sercos III 드라이브(실제 축) 추가

연결된 모든 드라이브에 대한 데이터를 확인하는 가장 간단한 방법은 sercos III 버스를 스캔하는 것입니다(상황에 맞는 메뉴 "sercos" 노드 버스 구성 스캔(Scan bus configuration)).

스캔을 위해 IndraWorks는 프롬프트를 표시한 후 컨트롤을 매개변수 입력 모드(P2)로 전환합니다.

결과는 아래와 같이 표시됩니다.

## 첫 단계 - 모션



위치(Pos.) 토폴로지 주소(링에서의 순서)  
 제품유형 (Device type) 드라이브 제품군  
 주소(Addr.) sercos 주소  
 번호(No.) 축 번호  
 드라이브 이름(Drive Name) 기본 이름: 드라이브 n, 이름이 바뀜  
 이름(Name) 축 / I/O 제품의 이름  
 펌웨어 (Firmware) 펌웨어 ID  
 IPO 컨트롤 또는 드라이브의 보간(이 경우 컨트롤)  
 그림 5-10: sercos 드라이브 목록(드라이브 이름이 RA1로 변경됨)



컨트롤 [Icon] 또는 드라이브 [Icon]의 보간:  
 아이콘을 두 번 클릭하여 전환합니다.

"드라이브 만들기"(Create Drives)를 클릭합니다.

스캔된 버스 구성이 프로젝트 탐색기로 전송됩니다. 드라이브는 모션 노드에서 "실제 축"(Real Axis)으로 표시됩니다.

축 이름 "RA1" 대신 축 번호 "1"을 사용할 수 있습니다.

프로젝트에는 "IndraLogic XLC L65" 유형의 "DCC\_Control" 제품과 실제 축 "RA1"(폴더 모션(Motion) ▶ 실제 축(Real Axes)) 또는 드라이브 "RA1"(폴더 sercos)이 포함되어 있습니다.

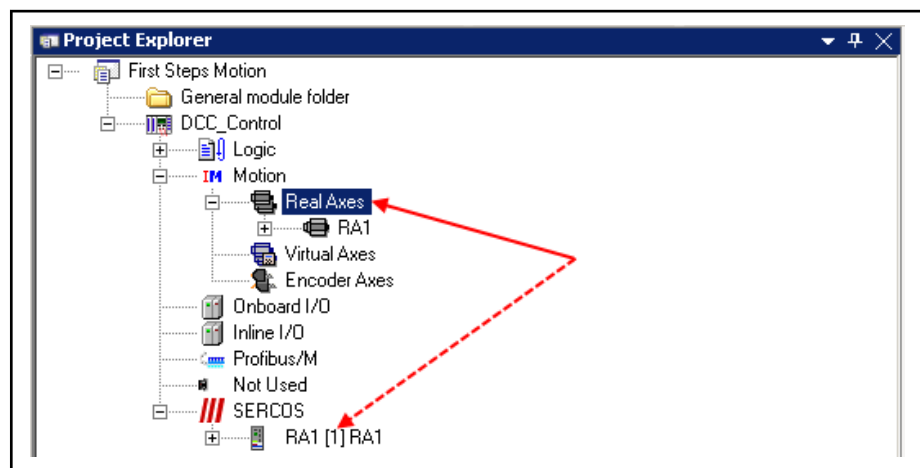


그림 5-11: 드라이브/실제 축 "RA1"이 프로젝트에 추가됨

### 5.3.7 프로젝트에 가상 축 추가

다양한 인터페이스(예: sercos 또는 PROFIBUS)에서 실행할 수 있어 그에 따라 선언하거나 스캔해야 하는 실제 축과 달리 가상 축은 선언하기만 하면 됩니다.

## 첫 단계 - 모션

다. 프로젝트 탐색기에서 **모션(Motion) > 가상 축(Virtual Axes)** 폴더를 마우스 오른쪽 버튼으로 클릭하고 상황에 맞는 메뉴 항목 **추가(Add)**를 선택합니다.

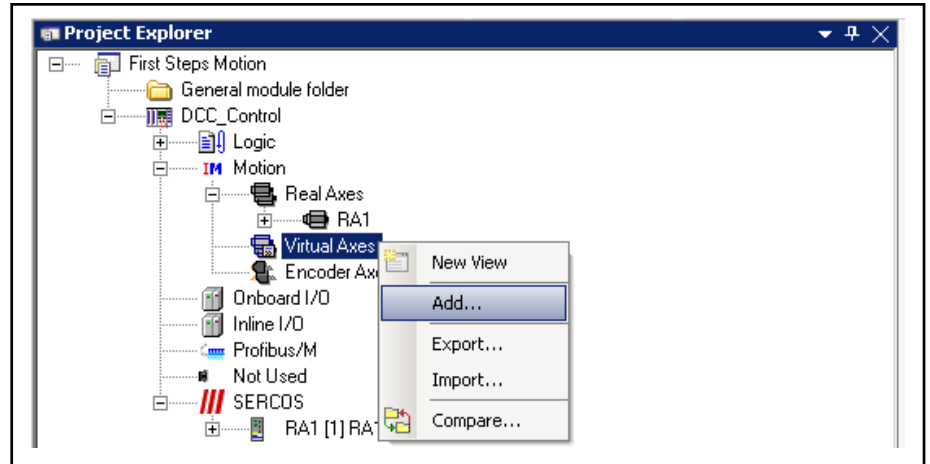


그림 5-12: 가상 축 선언

컨트롤 입력과 마찬가지로 마법사는 정보를 입력하는 데 필요한 대화 상자를 제공합니다.

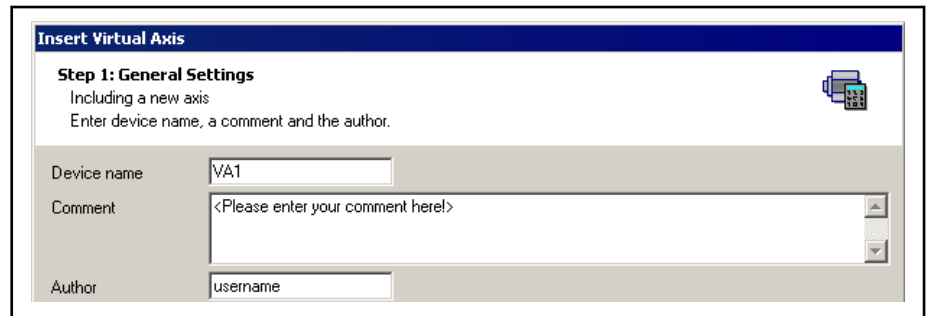


그림 5-13: 가상 축의 "일반 설정"(General Settings) 대화 상자

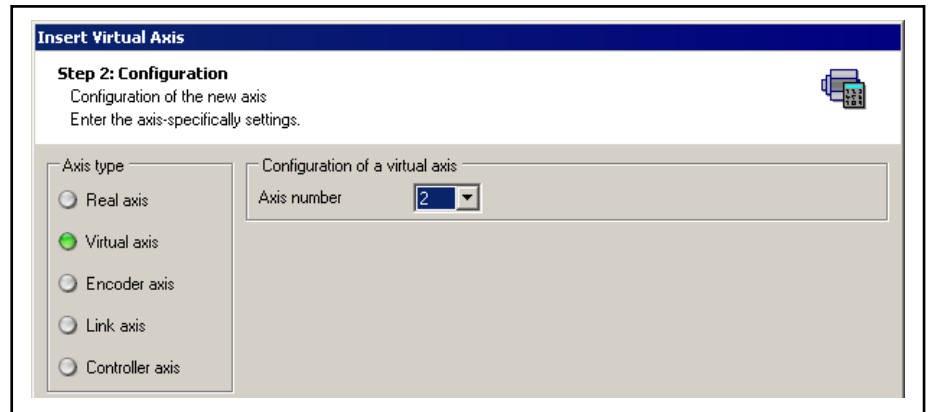


그림 5-14: 가상 축의 "구성"(Configuration) 대화 상자

축 이름 "VA1" 대신 축 번호 "2"를 사용할 수 있습니다.  
"완료"(Finish)를 클릭합니다.

### 5.3.8 모션 하위 프로젝트 다운로드

첫 번째 편집 작업은 컨트롤과 "첫 단계 - 모션"(First Steps - Motion) 프로젝트의 모든 축을 만들고 컨트롤 및 축 구성 요소의 매개변수를 입력하여 완료되었습니다.

## 첫 단계 - 모션

이제 지금까지 수집된 정보를 컨트롤로 로드해야 합니다. 원하는 제품(DCC\_Control)를 강조하고 프로젝트 도구 모음의 아이콘 또는 메뉴 항목 **프로젝트(Project) ▶ 제품을 온라인으로 전환(Switch Device Online)**을 클릭합니다.

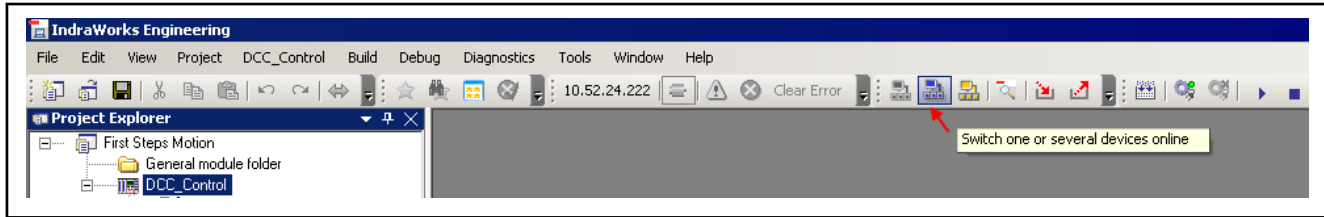
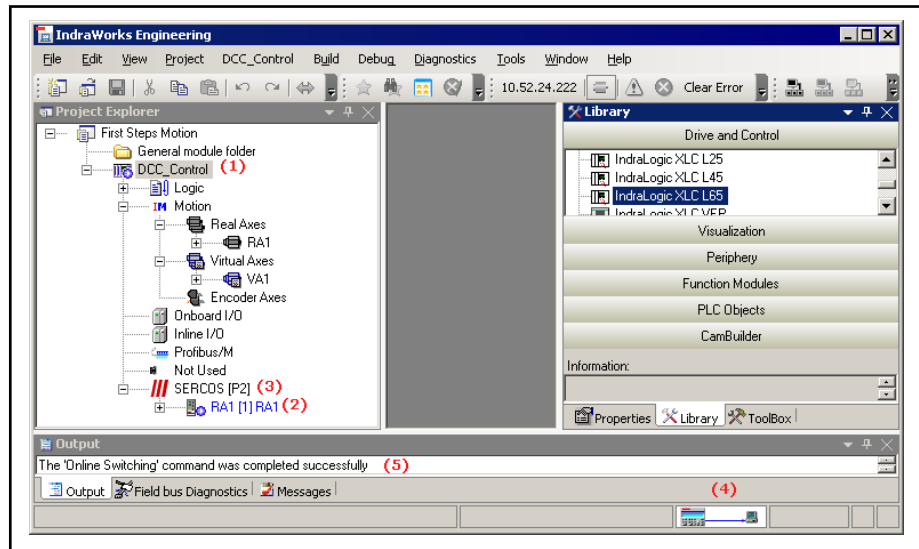


그림 5-15: 모션 데이터를 컨트롤로 로드

컨트롤의 프로젝트와 자체 프로젝트 간 차이점에 대한 정보가 표시됩니다. 차이점을 수락하면 덮어쓰기가 실행됩니다. 컨트롤 화면 표시가 일련의 단계에서 **BB 중지(BB STOP)**에서 **P2 중지(P2 STOP)**로 바뀝니다.



이 다운로드를 진행하지 않으면 백그라운드로 처리된 데이터가 컨트롤에서 누락됩니다.



- (1) 활성 컨트롤(온라인)
- (2) 활성 드라이브(온라인)
- (3) "P2" 수준에서의 통신, 매개변수 입력 모드
- (4) 온라인 정보
- (5) 상태 표시줄

그림 5-16: 다운로드가 성공적으로 완료됨(상태 표시줄)



바닥글의 "온라인" 정보(4)는 원하는 컨트롤 또는 해당 하위 폴더 중 하나가 활성 상태일 경우에만 표시됩니다.



BB로 전환(작동 준비)하기 전에 P2에서 드라이브(축)에 대해 적절한 매개변수 값을 입력해야 합니다. 그렇지 않으면 단계 전환 오류가 발생하여 BB 단계에 도달할 수 없으며 다른 단계에서 P2로 전환하도록 해야 합니다(단계 전환 중에 드라이브 및 컨트롤에서 적합성 검사).

이는 일반적으로 컨트롤을 BB로 전환하거나 드라이브를 Ab(드라이브 준비)로 전환하려고 하는 경우 적용됩니다.



드라이브의 작동 모드가 IndraDrive(드라이브에서 계산)에서 IndraDrive(컨트롤에서 계산)로 전환되었거나 그 반대로 전환된 경우 P2 에서 BB 로 변경할 때 오류 메시지가 발생할 수 있습니다. 이 경우 드라이브의 기본 매개변수를 다시 로드해야 합니다. RA1 의 상황에 맞는 메뉴 **매개변수(Parameters) ▶ 기본 매개변수 로드... (Load basic parameters...)**

그런 다음 컨트롤을 "매개변수 입력 모드(P2)"에서 "작동 모드(BB)"로 전환합니다(sercos 노드의 상황에 맞는 메뉴 **sercos 상태(sercos state) ▶ 단계 P4(Phase P4)**).



인코더의 유형에 따라 "F2174, 인코더에 의한 참조 손실" 오류가 발생할 수 있습니다.

오류는 드라이브 및 컨트롤의 디스플레이에서 <ESC> 키를 통해 확인할 수 있으며 **DCC\_Control ▶ 모션(Motion) ▶ 실제 축(Real Axis) ▶ RA1 ▶ 치수 참조 설정(Establish Dimensional Reference) ▶ 치수 참조 - 모터 인코더(Dimensional reference - motor encoder)** "절대 위치 설정"을 통해 제거할 수 있습니다.

드라이브는 이제 "Ab"(드라이브 준비) 상태가 됩니다.

컨트롤, "BB"의 모션 구성 요소가 작동할 준비가 되고 PLC 구성 요소는 "중지" 상태가 됩니다.

## 5.4 FBD(평선블록다이어그램)에서 모션 스텝스 프로그래밍

### 5.4.1 FBD(평선블록다이어그램)에서 모션 스텝스 프로그래밍, 일반 정보

PLC 프로그램에서 PLCopen 평선블록(EN 61131-3 에 따라 프로그래밍 시스템 IndraLogic 2G)을 사용하여 모션 스텝스를 프로그래밍합니다.

DDC\_Control 컨트롤, 실제 축 RA1 및 가상 축 VA1 을 사용하여 프로젝트를 만들 때 모션 스텝스를 프로그래밍하는 첫 번째 작업은 "로직"(Logic) 폴더 아래에서 시작되었습니다.

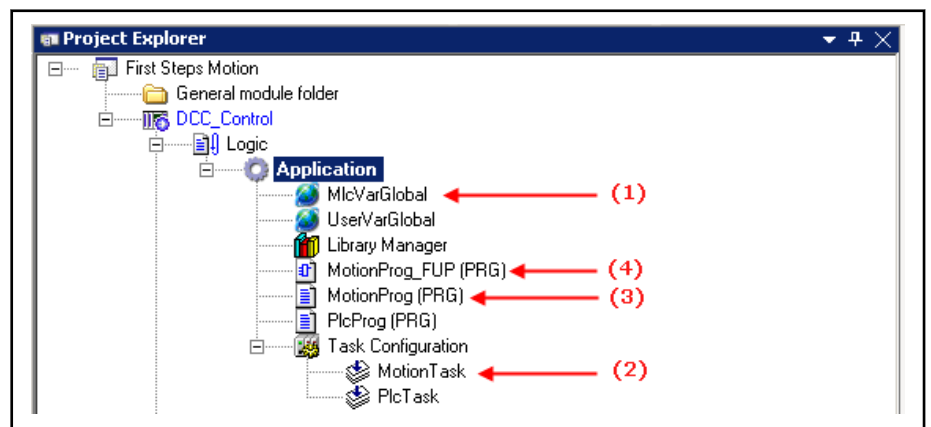


그림 5-17: "로직"(Logic) 노드를 확장한 후의 프로젝트 탐색기

다음 프로그램이 생성되었습니다.

- **MotionProg (3)**(ST(구조체언어) 사전 선택)
- **PlcProg**(ST(구조체언어) 사전 선택), 사용 안 함

세 번째 프로그램 **MotionProg\_FBD ( 4 )**는 응용 프로그램(Application)을 마우스 오른쪽 버튼으로 클릭한 후 **추가(Add) ▶ POU(프로그램 유닛)**에서 이름을

## 첫 단계 - 모션

입력하고 프로그래밍 언어로 FBD(평선블럭 다이어그램)를 선택하여 수동으로 추가했습니다.

프로그램은 현재 비어 있습니다.

"작업 구성"(Task Configuration) 아래에 다음 작업이 생성되었습니다.

- **MotionTask ( 2 )**, "MotionProg"의 경우 (우선 순위 2, sercos 작업에 의해 외부에서 이벤트 트리거됨, 위치독 활성)  
이 작업 아래에서 실행 중인 프로그램은 sercos 주기의 지정된 파트 내에 완전히 처리해야 합니다. 이러한 프로그램이 이 파트 내에 없으면 컨트롤이 중지됩니다.
- **PlcTask**, PlcProg 의 경우 (낮은 우선 순위 주기 작업), 사용 안 함  
남은 시간에 프로그램을 조금씩 처리합니다. 즉, 필요한 경우 여러 주기 동안 처리됩니다.

다음에 대해 전역 변수/전역 상수가 생성되었습니다.

- **UserVarGlobal**: 사용자에게 대해 예약됨, "첫 단계 - 모션"(First Steps - Motion)에서 사용 안 함
- **MlcVarGlobal ( 1 )**: RA1 및 VA1 축에 대한 주소 상수 포함

## 프로그램

```
(* Automated generated code by MLC. *)
(* Please don't edit, the code will be overwritten...*)
VAR_GLOBAL CONSTANT
(*virtual axis "VA1" with axis number 2*)
VA1: AXIS_REF :=(CntlNo:=LOCAL_CNTRL,AxisNo:=AXIS_2);
(*real axis "RA1" with axis number 1*)
RA1: AXIS_REF :=(CntlNo:=LOCAL_CNTRL,AxisNo:=AXIS_1);
(*...*)
END_VAR
```



"첫 단계 - 모션"(First Steps - Motion)에는 MotionTask 를 사용합니다.

필요한 프로그래밍 언어에 따라 **MotionProg** 또는 **MotionProg\_FBD** 가 할당됩니다.

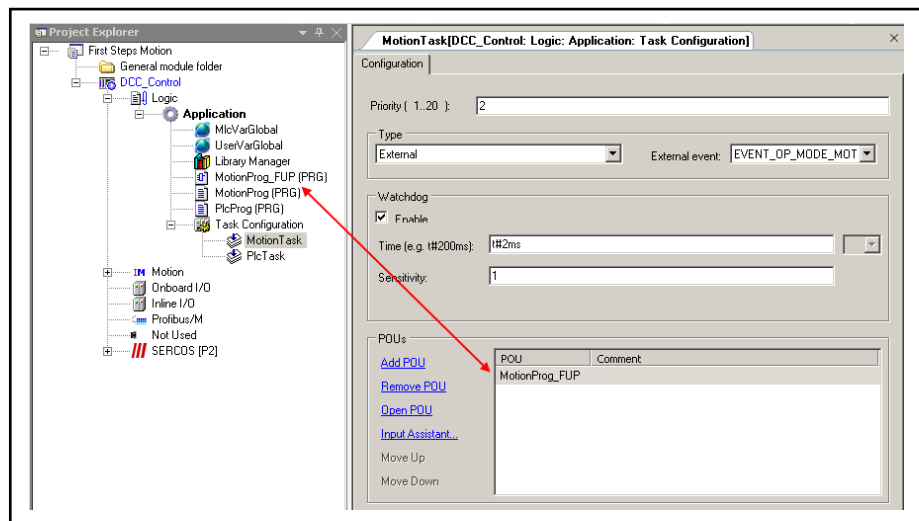


그림 5-18: 호출 삭제: MotionProg. 호출 추가: MotionProg\_FBD

## 5.4.2 가상 축 VA1 프로그래밍, 프로그래밍 언어 FBD(평선블럭다이어그램)



FBD(평선블럭다이어그램)에서 프로그래밍을 그래픽으로 표시하려면 이 섹션의 내용을 계속해서 살펴보십시오.

ST(구조체언어)를 사용해야 할 경우 [가상 축 VA1 프로그래밍, 프로그래밍 언어 ST\(구조체언어\)](#) 섹션, 117 페이지를 참조하십시오.

프로젝트 목표 첫 단계 - 모션, 81 페이지와 마찬가지로 PLCopen 블럭 MC\_MoveVelocity 및 MC\_Stop 의 인스턴스는 "VA1"을 사용하여 호출합니다.



가상 축에는 MC\_Power 인스턴스가 필요하지 않습니다.

아래에 설명된 대로 진행합니다.

1. "MotionProg\_FBD" 프로그램에 "MotionTask"를 할당합니다.
2. "MotionProg\_FBD" 프로그램을 두 번 클릭하여 엽니다.

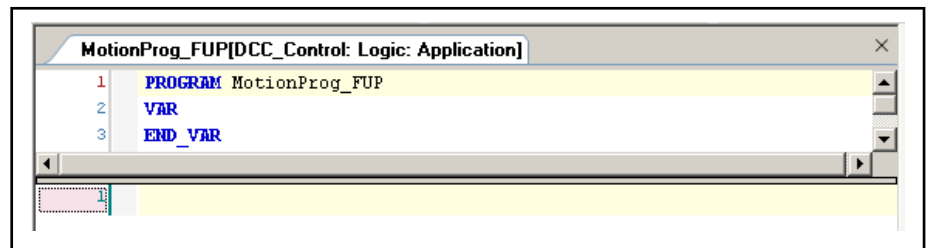


그림 5-19: FBD(평선블럭다이어그램)의 새 프로그램

첫 번째 평선블럭을 마우스 오른쪽 버튼으로 클릭하여 구현, 네트워크 1 에 삽입합니다.

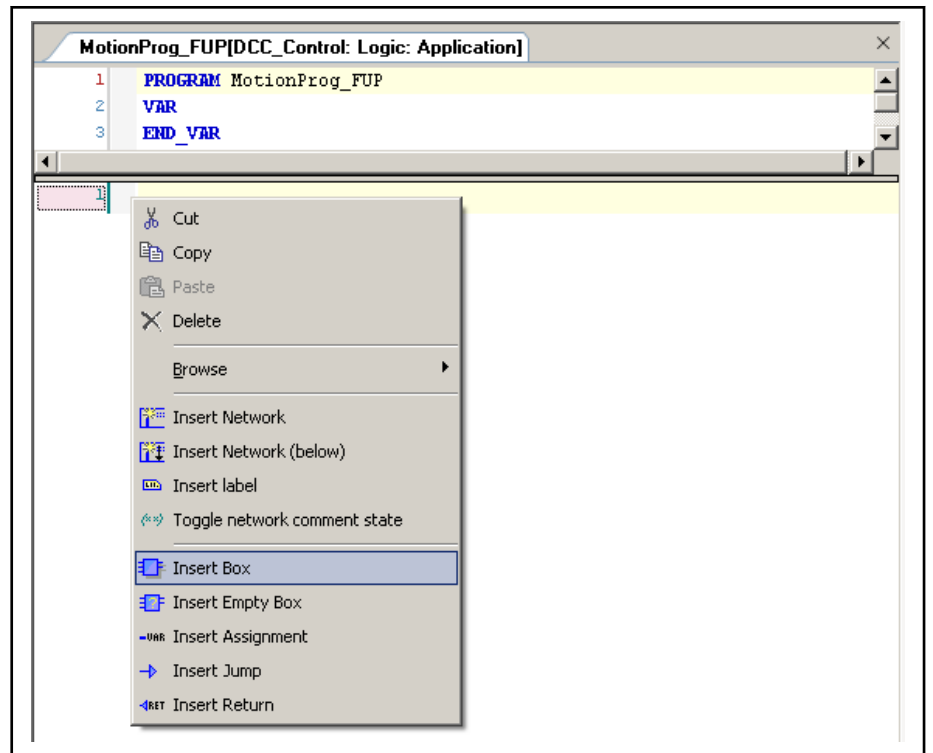


그림 5-20: 새 평선블럭 생성

이렇게 하려면 열린 선택 창에서 다음 항목을 선택합니다.

## 첫 단계 - 모션

평선블럭(Function Blocks) ▶ ML\_PLCPEN ▶ POU(프로그램 유닛) ▶ 모션(Motion) ▶ MC\_MoveVelocity

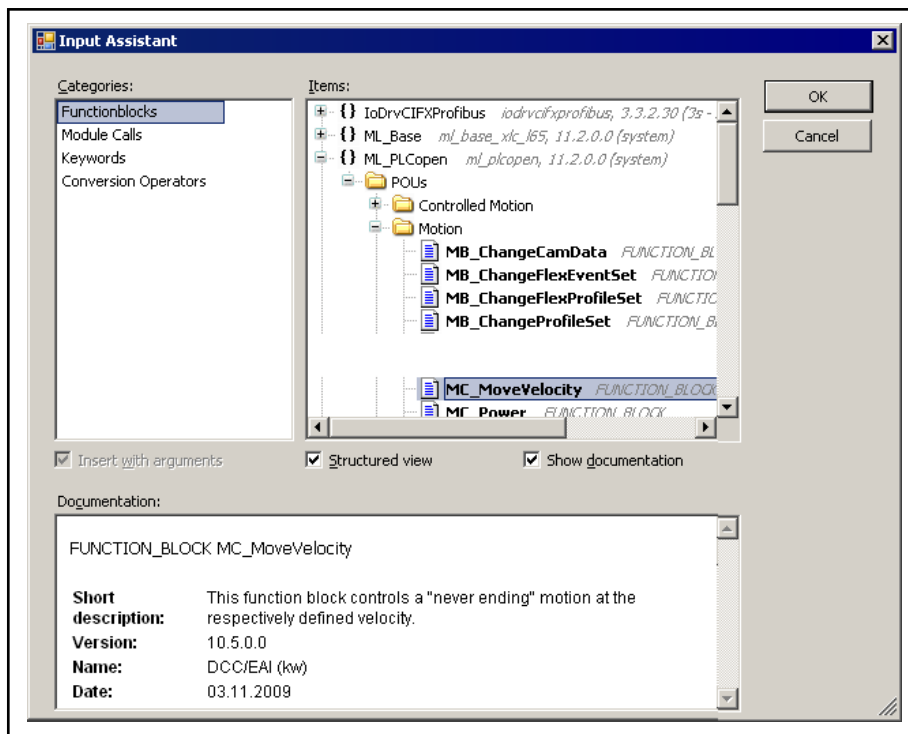


그림 5-21: 평선블럭 유형의 선택 창

"확인"(OK)을 클릭하여 MC\_MoveVelocity 평선블럭의 삽입을 확인하고

"MotionProg\_FUP" 프로그램에서 3 개의 물음표 ??? 을 마우스 오른쪽 버튼으로 클릭하여 강조합니다. 이 "MoveVelVA1" 평선블럭의 인스턴스 이름을 입력하고 <Enter> 키를 눌러 입력을 확인합니다. "확인"(OK)을 클릭하여 바로 승인할 수 있는 "변수 선언" 창이 열립니다.

첫 단계 - 모션

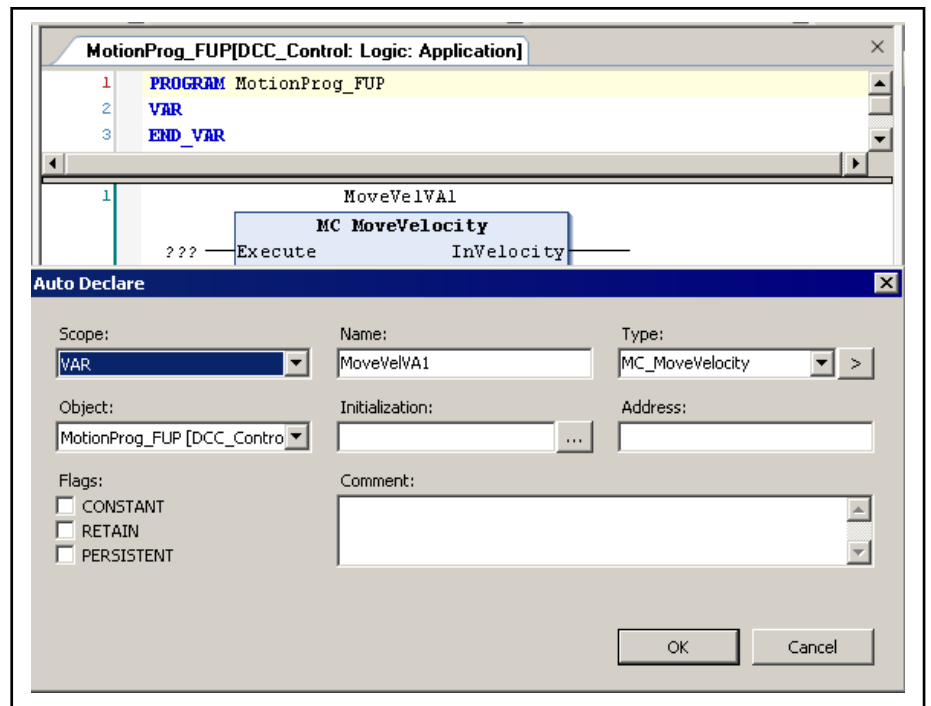


그림 5-22: "변수 선언" 창을 사용하여 "MC\_MoveVelocity" 유형의 FB(평선블럭)의 인스턴스 이름 선언

이제 PLCopen FB MC\_MoveVelocity의 MoveVelVA1 인스턴스가 선언되었습니다. FB(평선블럭) 입력을 할당합니다.

FB(평선블럭)의 인스턴스 이름과 동시에 BOOL 유형의 "ex\_MoveVelVA1" 변수를 사용하여 "Execute" 입력을 할당합니다.

- "Execute" 입력 앞에 있는 3 개의 물음표를 강조합니다.
- 이름 "ex\_MoveVelVA1"을 입력하고 <Enter> 키를 눌러 완료합니다.
- 열리는 창 "변수 선언"은 <Enter> 키를 눌러 바로 확인할 수 있습니다.

"Velocity" 입력을 할당하려면 다음 단계를 진행합니다.

- "Velocity" 입력 앞에 있는 3 개의 물음표를 강조합니다.
- 이름 "VcomVA1"을 입력하고 <Enter> 키를 누릅니다.
- 유형으로 REAL 을 입력하고 초기 값으로 200(rpm)을 입력합니다. <Enter> 키를 누릅니다.

"Acceleration" 입력으로 20(rad/s²)을 입력하고 "Deceleration"에 20(rad/s²)을 입력하십시오.



FB(평선블럭) 입력에 바로 값을 입력합니다.

또는 개별 값을 갖는 변수를 사용할 수 있습니다.

"Axis" 입력을 할당하려면 다음 단계를 진행합니다.

- ??? 을 클릭하고 가까이 표시된 입력 보조를 ... 을 선택합니다.

변수(Variables) ▶ 응용 프로그램(Application) ▶ MlcVarGlobal ▶ VA1 을 선택합니다.

## 첫 단계 - 모션

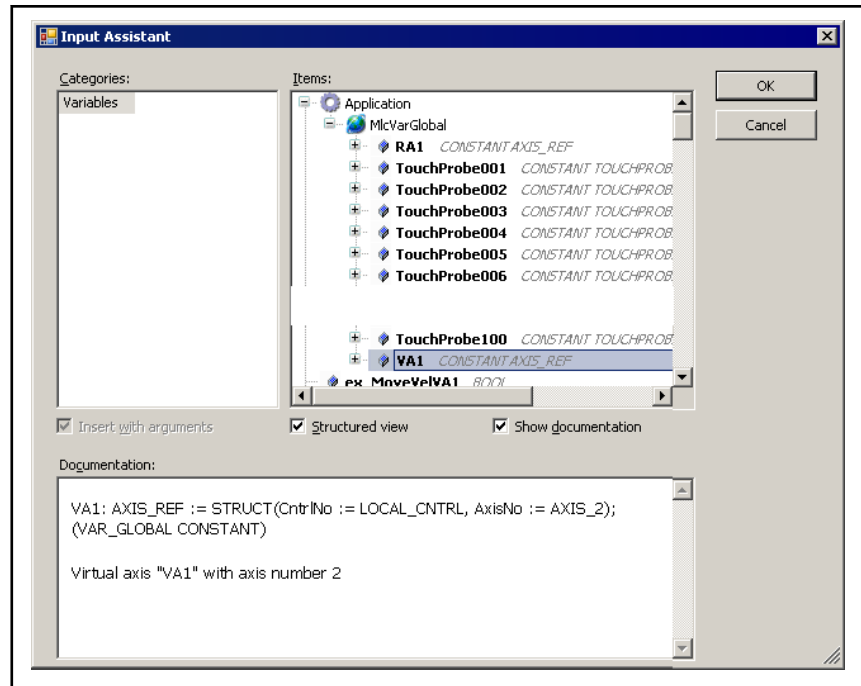


그림 5-23: "MC\_MoveVelocity" 유형의 평선블럭을 할당할 때 축 인스턴스 선택

- "확인"(OK)을 클릭하여 확인합니다.
- <Enter> 키를 눌러 이름 입력을 완료합니다.



전역 상수의 이름은 다음과 같이 폴더 이름과 요소 이름으로 구성됩니다.

MlcVarGlobal.VA1

평선블럭이 제대로 작동하는지 여부에 대한 피드백을 받으려면 해당 출력을 할당합니다.

마우스를 사용하여 "InVelocity" 출력 옆의 오른쪽 영역을 강조하고 마우스 오른쪽 버튼을 클릭하여 상황에 맞는 메뉴에서 다음을 선택합니다.

**할당 삽입**

첫 단계 - 모션

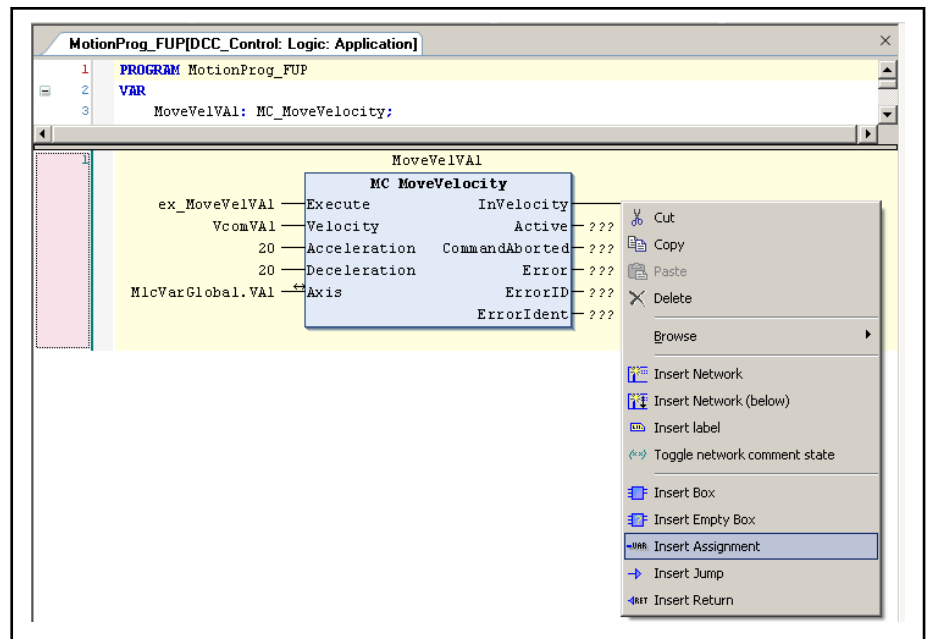


그림 5-24: 평선블럭의 출력 할당

BOOL 유형의 "in\_MoveVelVA1" 변수를 사용하여 블럭의 "InVelocity" 출력을 할당합니다.

- "InVelocity" 출력에서 3 개의 물음표 ??? 을 강조합니다.
- 이름 "in\_MoveVelVA1"을 입력한 다음 <Enter> 키를 누릅니다.
- 열려 있는 "변수 선언" 창은 <Enter> 키를 눌러 바로 확인할 수 있습니다.
- "Active" 출력에 "ca\_MoveVelVA1"을 할당합니다.
- "CommandAborted" 출력에 "ca\_MoveVelVA1"을 할당합니다.
- "Error" 출력에 "er\_MoveVelVA1"을 할당합니다.
- "ErrorID" 출력에 "ID\_MoveVelVA1"을 할당합니다.
- "Error" 출력에 "Ident\_MoveVelVA1"을 할당합니다.

MC\_MoveVelocity 유형의 PLCopen FB 가 해당 작업에 따라 할당됩니다. 선언 영역이 자동으로 완성되었습니다.

## 첫 단계 - 모션

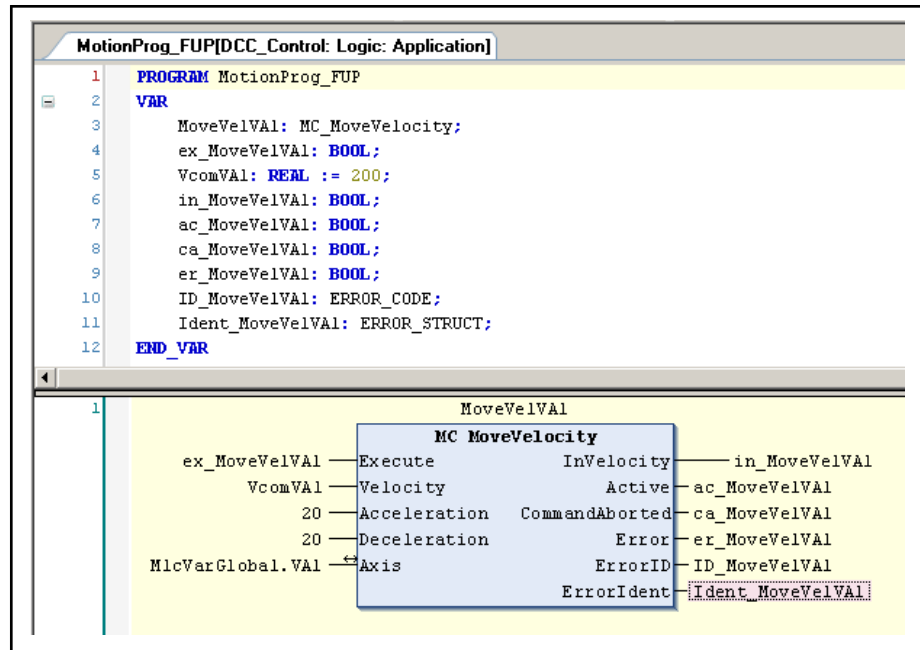


그림 5-25: "MC\_MoveVelocity" 유형의 PLCopen FB 가 삽입 및 할당됨



평선블럭이 완료될 때마다 <Shift>+<F11>을 사용하여 테스트 컴파일을 실행합니다. 오류가 있으면 바로 지우십시오.

PLCopen FB MC\_Stop 의 인스턴스를 생성합니다.

### MC\_Stop MC\_Stop 삽입 및 "StopVA1" 인스턴스 선언

새 네트워크를 삽입하려면 마우스 오른쪽 버튼으로 네트워크 1 을 클릭합니다.

#### 네트워크 삽입(아래)

MC\_Stop 삽입 및 "StopVA1" 인스턴스 선언

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Execute" → BOOL 유형의 "ex\_StopVA1"

"Deceleration" → 50

"Axis" → "AXIS\_REF" 유형의 "MlcVarGlobal.Val1"

출력:

"Done" → BOOL 유형의 "do\_StopVA1"

"Active" → BOOL 유형의 "ac\_StopVA1"

"CommandAborted" → BOOL 유형의 "ca\_StopVA1"

"Error" → BOOL 유형의 "er\_StopVA1"

"ErrorID" → ERROR\_CODE 유형의 "ID\_StopVA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "Ident\_StopVA1"

**실제 속도 "VactVA1"** 마지막으로 축 VA1 "VactVA1"(데이터 유형 REAL)의 현재 속도를 읽을 수 있는 네트워크를 구현합니다.

현재 속도는 축의 데이터 집합(공유 메모리)에서 읽어 옵니다.

MLC 를 제어할 수 있는 모든 축의 데이터는 다음 배열에 있습니다.

**GVL\_Base.AxisData: ARRAY [1..xx<sup>1</sup>)] OF ML\_AXISDATA\_SM**

## 첫 단계 - 모션

여기에서 축 번호는 색인으로 사용됩니다.

변수의 이름을 다음과 같이 지정합니다.

**GVL\_Base.AxisData[2].rActualVelocity\_i,**

가상 축의 경우 축 번호는 2 입니다. VA1 을 선언하기 전에 RA1 스캔이 실행되었으므로 축 번호 1 은 RA1 입니다.

- 새 네트워크를 삽입하려면 마우스 오른쪽 버튼으로 네트워크 2 을 클릭합니다.

**네트워크 삽입(아래)**

- **할당 삽입**
- 오른쪽의 ??? 에 "VactVA1"을 삽입하고 <Enter> 키를 누른 다음 선언창에서 "REAL" 유형을 선택하고 <Enter> 키를 누릅니다.
- 왼쪽의 ??? 으로 이동한 후 [...] 을 클릭하고 입력 보조툴을 사용하여 변수(Variables) ▶ ML\_BASE ▶ 전역변수리스트(Global Variable List) ▶ GVL\_Base ▶ AxisData 로 이동합니다.

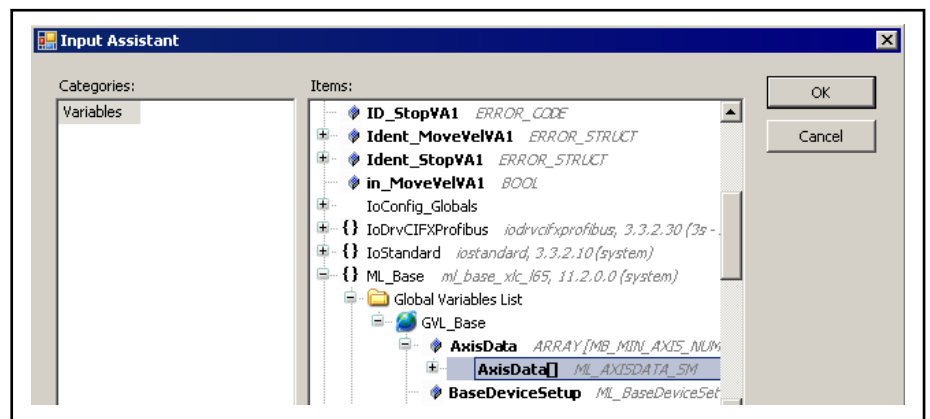


그림 5-26: ML\_Base.library 라이브러리의 전역변수리스트에 있는 실제 속도 값 VA1

Intellisense 기능을 통해 추가로 입력할 수 있습니다. 즉, "GVL\_Base.AxisData[2]."에 대해 "."을 입력한 후 사용 가능한 모든 요소를 열어 선택할 수 있습니다.

<Enter> 키를 눌러 "rActualVelocity\_i"를 확인합니다.

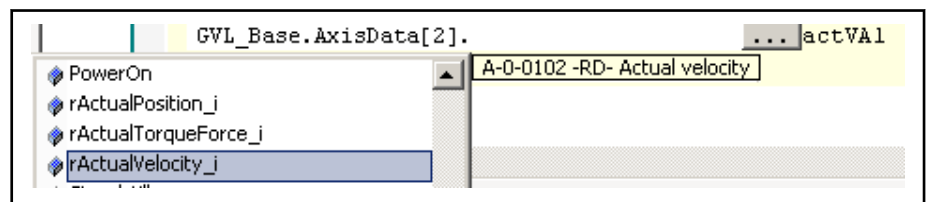


그림 5-27: "ActualVelocity" 변수 선택

작업 할당과 관련된 모션 스퀀스의 구현이 다음과 같이 완료되었습니다.

1) 제어할 수 있는 축의 개수는 컨트롤 유형 및 해당 펌웨어에 따라 달라집니다(현재 L65 의 경우: 축 64 개).

## 첫 단계 - 모션

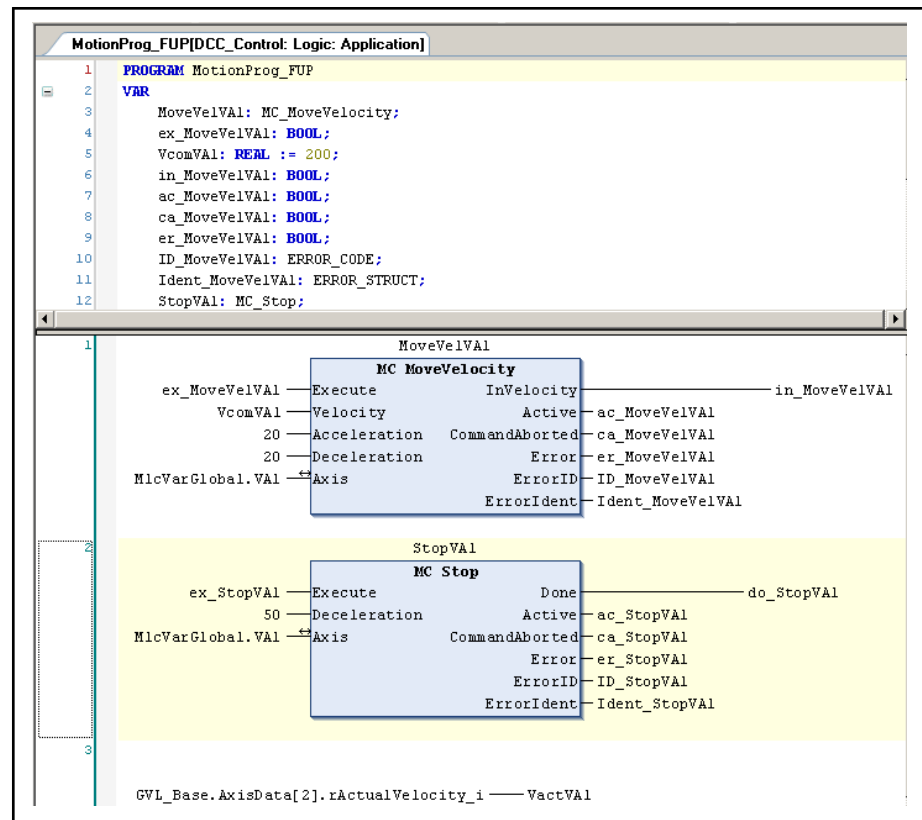


그림 5-28: 가상 축 VA1 구현

## 5.4.3 PLC 하위 프로젝트 가상 축 VA1 다운로드(FBD(평선블럭다이아그램))

위에서 구현된 모션 스텝스는 컴파일하고 컨트롤에 적용한 후 시작해야 합니다.

다음은 통해 컴파일에 액세스할 수 있습니다.

- **만들기(Create) ▶ 작성(Build):** 응용 프로그램의 추가된 부분 컴파일
- **만들기(Create) ▶ 다시 작성(Rebuild):** 전체 응용 프로그램 컴파일

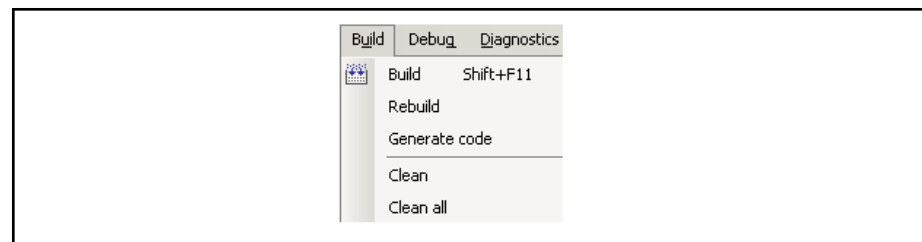


그림 5-29: 응용 프로그램 작성

컴파일된 데이터를 대상 컨트롤에 적용하려면 메뉴 항목 디버그(Debug) ▶ 로그인(Login)을 선택합니다. 아래 그림을 참조하십시오.

첫 단계 - 모션

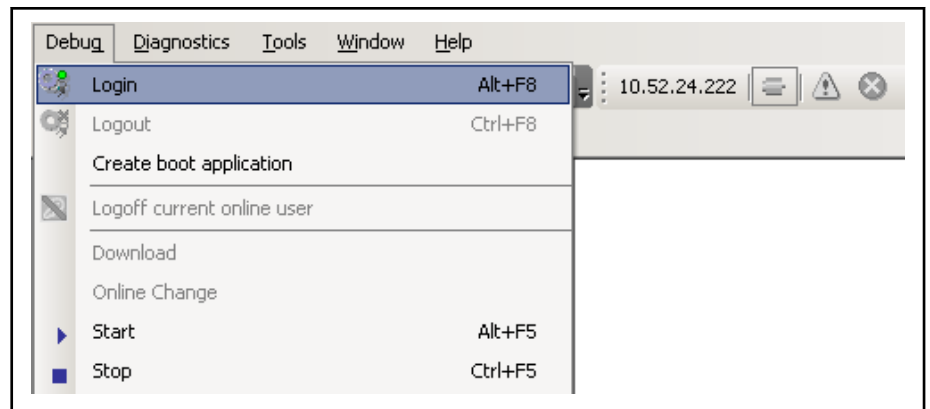


그림 5-30: 컴파일된 데이터를 컨트롤로 전송

응용 프로그램이 이미 로드되었는지 그리고 이 응용 프로그램이 현재 응용 프로그램의 이전 버전인지에 따라 다운로드 동의 여부를 묻는 메시지가 나타납니다.

응용 프로그램은 성공적인 전송 후에도 계속 중지되어 있으며(프로젝트 탐색기 및 상태 표시줄에 빨간색으로 표시됨) 값을 표시하도록 선언 및 구현에 필요한 변수가 준비됩니다. 그러나 값이 유효하지 않습니다.

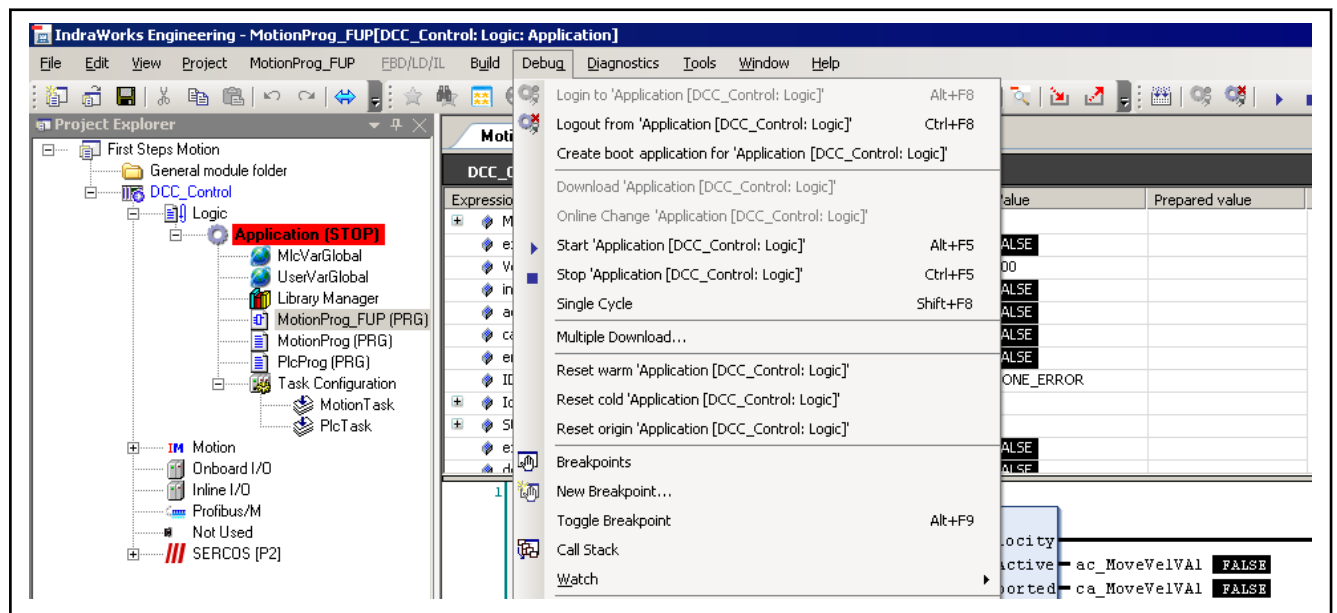


그림 5-31: 컨트롤에서 프로그램 처리 시작(1)

디버그(Debug) ▶ 시작(Start)을 통해 컨트롤에서 처리를 시작합니다.

## 첫 단계 - 모션

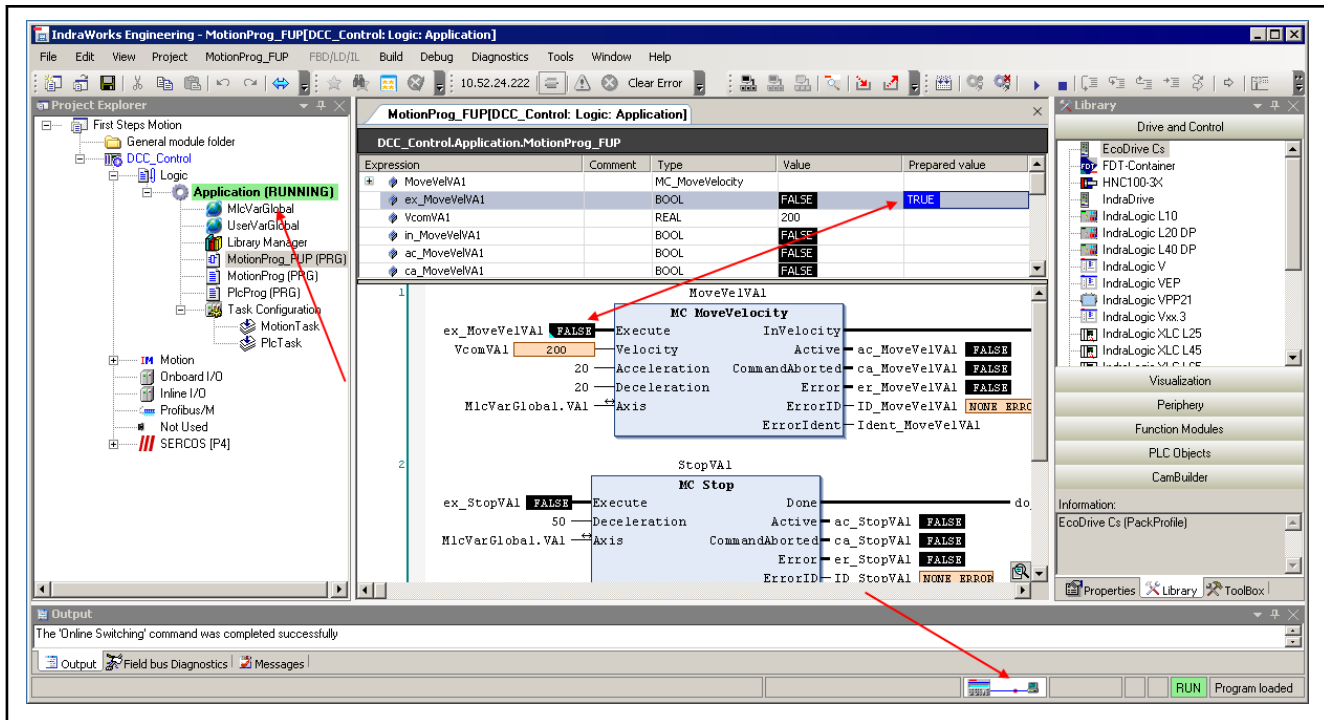


그림 5-32: 컨트롤에서 프로그램 처리 시작(2)

프로그래밍된 평선블럭의 처리가 차례로 활성화되면 모션 프로세스가 자동으로 시작됩니다.

## 입력 변수의 "값 쓰기"

"MC\_MoveVelocity" 유형의 "MoveVelVA1"을 처리하려면 FB(평선블럭)의 "Execute" 입력 시 상승 에지가 발생해야 합니다.

마우스 왼쪽 버튼을 사용하여 입력 변수 "ex\_MoveVelVA1" 뒤에 있는 FALSE를 두 번 클릭합니다. "ex\_MoveVelVA1" 뒤의 "FALSE"가 강조되어 있습니다. 먼저, 선언 줄의 "TRUE"가 강조된 후에 다음 클릭 시 "FALSE"가 강조되고 빈 공간이 강조됩니다(비활성화됨).

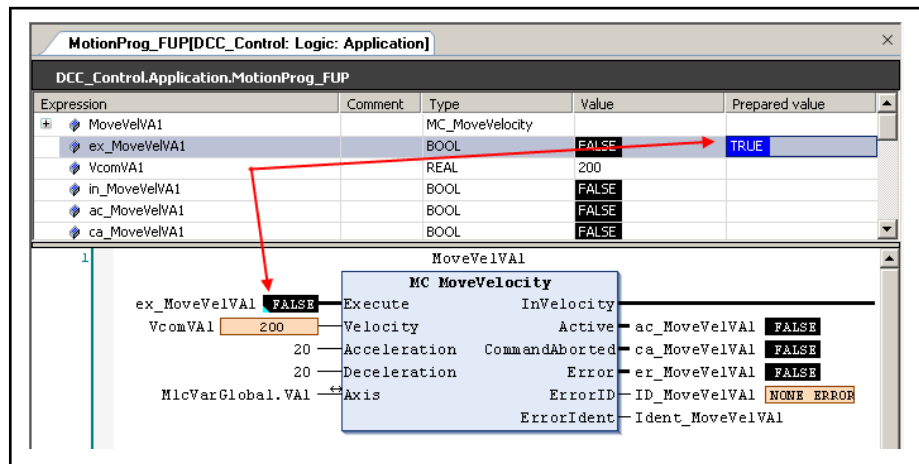


그림 5-33: "값 쓰기"(Write values) 준비

컨트롤에서 디버그(Debug) ▶ 값 쓰기(Write Values) "Ctrl"+"F7"을 통해 <ex\_MoveVelVA1>을 <TRUE>로 설정할 수 있습니다.

첫 단계 - 모션



- "값 쓰기"(Write Values)는 다음 주기가 시작할 때 변수를 원하는 값으로 **한 번** 설정하는 것을 의미합니다.
- "값 강제 적용"(Force Values)은 다음 주기가 시작할 때 변수를 원하는 값으로 **영구적으로** 설정하는 것을 의미합니다.

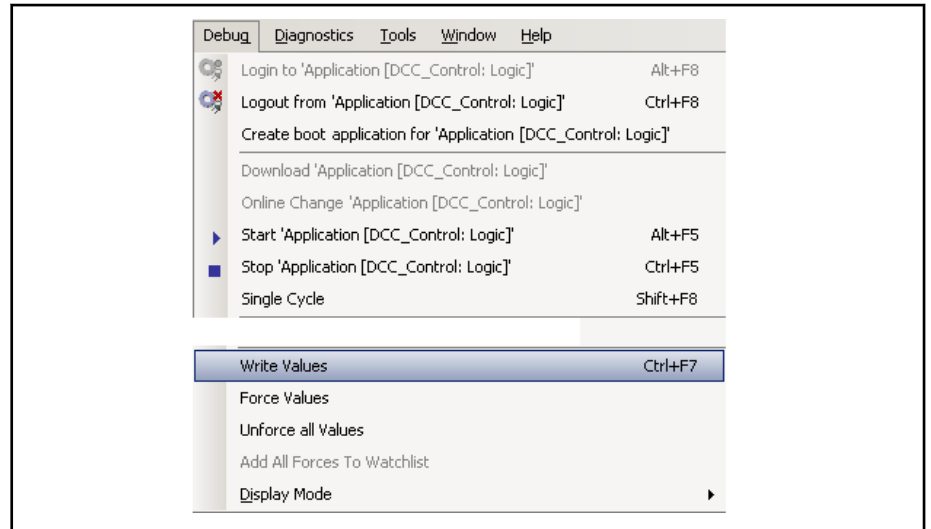


그림 5-34: 값 쓰기

생성된 상승 예지는 "MoveVelVA1" 평선블럭의 "InVelocity" 및 "Active" 출력을 "TRUE"①로 설정합니다.

속은 200rpm②으로 이동합니다.

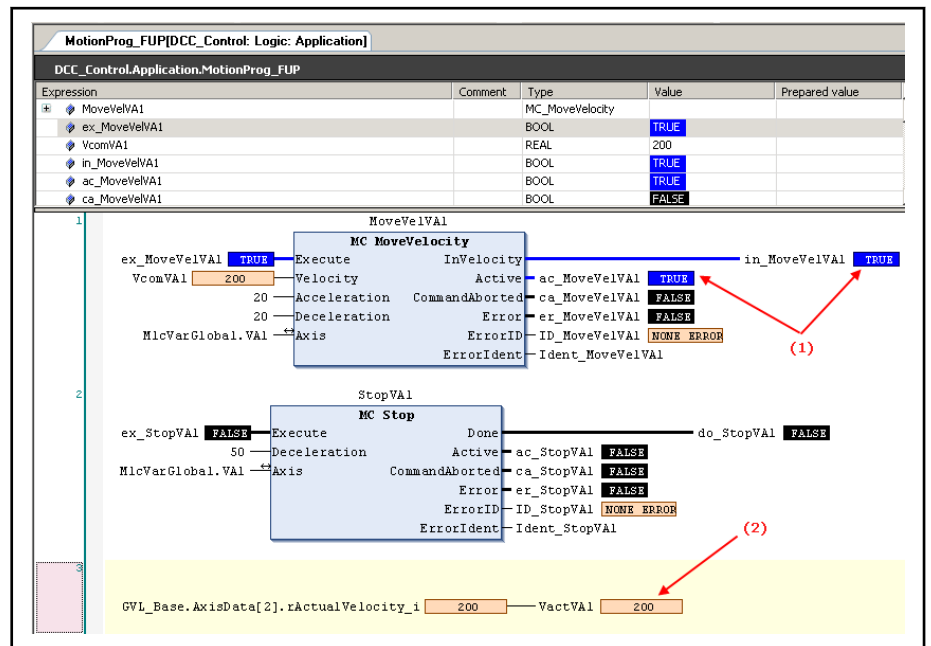


그림 5-35: "Execute" 입력의 "값 쓰기"(Write Values) 후

"ex\_MoveVelVA1"을 다시 "FALSE"로 전환합니다. 속은 계속해서 200rpm②으로 이동합니다.

MC\_Stop 유형의 "StopVA1" 평선블럭의 "Execute" 입력에서 "x\_StopVA1" 변수를 설정하여 축 모션을 중지할 수 있습니다.

응용 프로그램을 중지하고 로그아웃합니다.

## 첫 단계 - 모션

## 5.4.4 축 VA1의 모션 기록(트레이스), FBD(평선블록다이어그램)

IndraLogic은 모션 프로세스를 트레이스할 수 있는 트레이스 도구를 제공합니다. 트레이스 도구를 사용하여 다음 항목을 기록할 수 있습니다.

- 축 속도 "VactVA1"
- "MoveVelVA1"(MC\_MoveVelocity)의 경우
  - "ex\_MoveVelVA1"(Execute)
  - "in\_MoveVelVA1"(InVelocity)
  - "ca\_MoveVelVA1"(CommandAborted)

및

- "StopVA1"(MC\_Stop)의 경우
  - "ex\_StopVA1"(Execute)

이렇게 하려면 먼저 트레이스 기록을 구성해야 합니다.

**응용 프로그램(Application)**의 상황에 맞는 메뉴 **추가(Add) ▶ 트레이스...(Trace...)**을 통해 "Trace\_VA1" 개체를 삽입합니다. 개체를 두 번 클릭하여 엽니다.

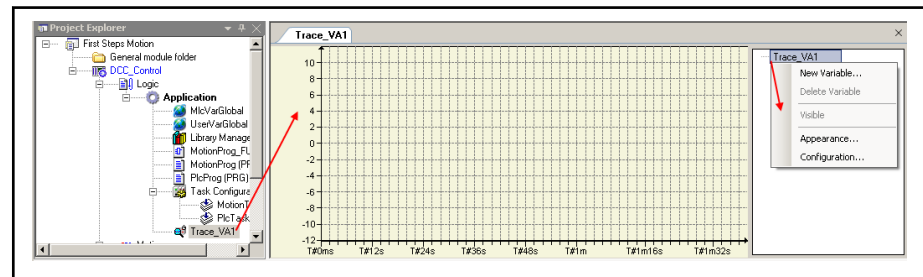


그림 5-36: 대화 상자: 상황에 맞는 메뉴가 표시된 새 트레이스 객체 "Trace\_VA1"

페이지의 오른쪽에서 상황에 맞는 메뉴를 통해 트레이스 객체와 관련하여 다음 작업을 수행할 수 있습니다.

- 구성...(Configuration...)
- 표시...(Appearance...)
- 새 변수...(New Variable...)

**구성...(Configuration...)** 먼저, 다음과 같이 새 트레이스 객체를 구성해야 합니다.

첫 단계 - 모션

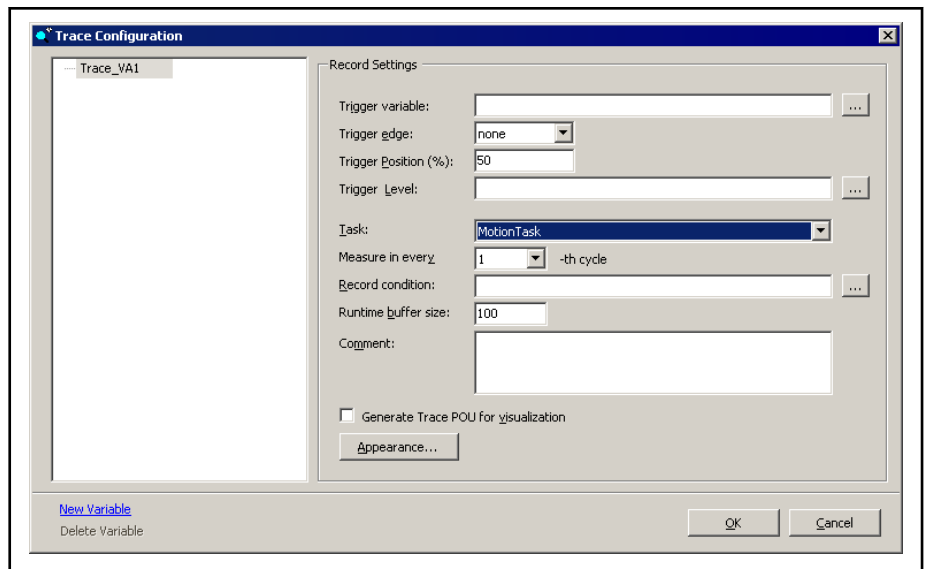


그림 5-37: 대화 상자: 새 트레이스 객체 "Trace\_VA1", 최소 구성

필요한 설정

- 작업: **MotionTask**
- 20 번째 주기마다 기록 즉,  $20 \times 2\text{ms} = 40\text{ms}$

**작업:(Task:)** 사용 가능한 작업 목록에서 처리한 후에 트레이스 변수의 값을 읽을 작업을 선택합니다. 기본값: 작업 구성 트리의 첫 번째 작업

**다음 주기마다 측정 ... 번째 주기:(Measure in every ... -th cycle:)** 트레이스 변수의 값을 다시 읽기 전에 경과할 주기 횟수를 입력합니다. (버퍼 크기 \* x 번째 주기마다 기록 \* 작업 간격) 수식을 사용하여 최소 기록 시간 프레임을 추정할 수 있습니다.

**기록 조건:(Record condition:)** 여기에 부울 변수, 값 또는 부울 식을 입력할 수 있습니다. 이 조건이 충족되는 즉시 TRUE 값을 받고, 값 기록이 시작됩니다. 아무 값도 입력하지 않으면 트레이스 구성이 컨트롤에 로드되고 응용 프로그램이 실행되는 즉시 값 기록이 시작됩니다.

**런타임 버퍼 크기:(Runtime buffer size:)** 현재 구성된 레코드에서 트레이스 변수 값을 수집하기 위해 런타임 시스템에 현재 제공된 링 버퍼의 크기(바이트)입니다. 그런 다음 트레이스 창에 값 곡선으로 버퍼 콘텐츠가 표시됩니다. 기본값: 100.



구성이 변경되면 트레이스 객체를 다시 기록해야 합니다.

디스플레이 설정(표시)...

이제 디스플레이 설정을 입력할 수 있습니다. 이러한 설정은 다시 로그인하지 않아도 유효합니다.

## 첫 단계 - 모션

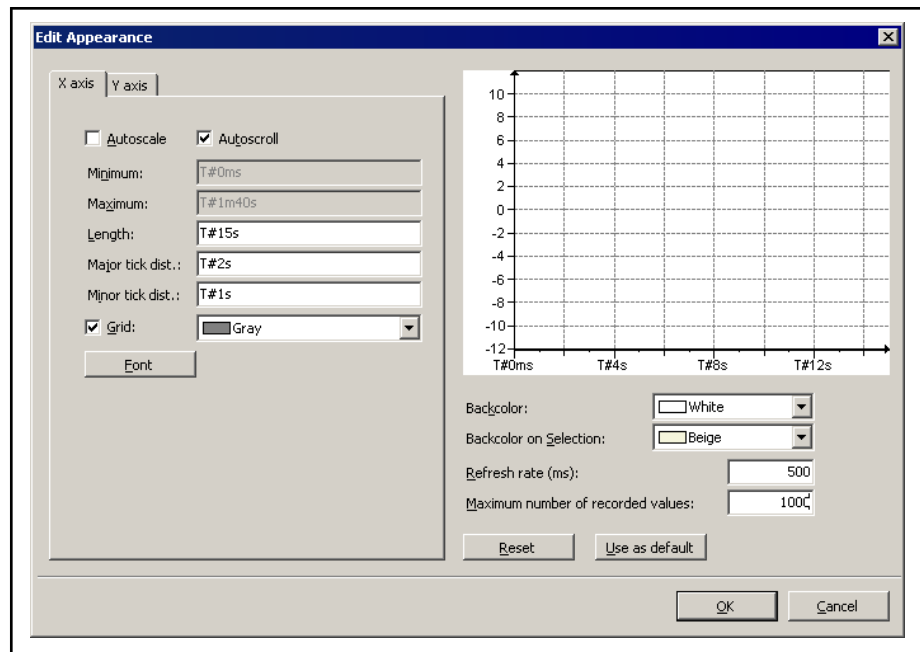


그림 5-38: 대화 상자: 디스플레이 설정(표시)

자동 스크롤(Autoscroll)을 선택하고 길이(Length)를 곡선에서 15 초 기간에 배치합니다. 스케일의 눈금폭이 1 초인 경우 스케일의 눈금폭이 2 초 그리드로 설정됩니다.



정밀한 프로세스를 위해서는 15 초를 초과하면 안 됩니다.

새 변수...

이제 필수 변수의 이름을 지정하고 이러한 변수의 화면 표시를 조정해야 합니다.

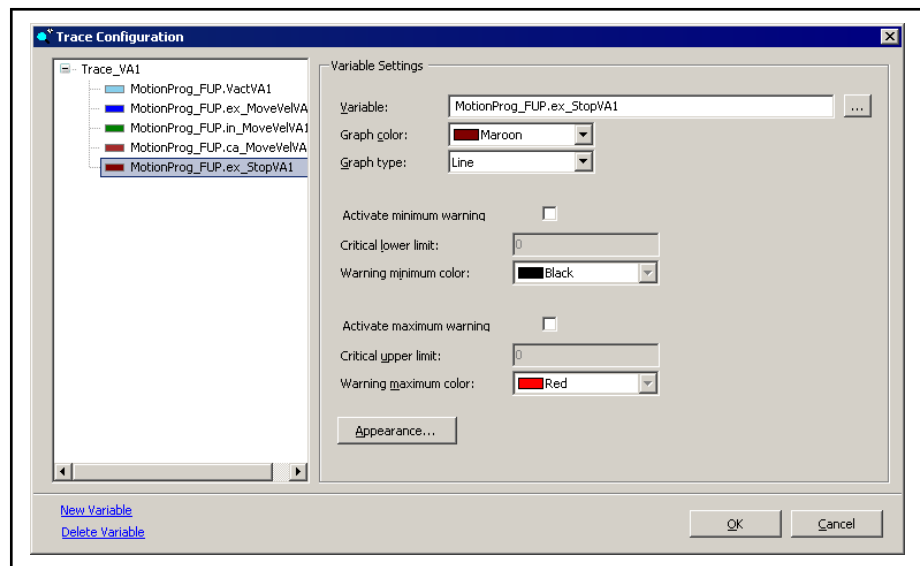


그림 5-39: 대화 상자: 새 변수 만들기 및 화면 표시 조정

색(Color) 및 그래프 유형(Graph type)은 변수(Variables) 곡선과 관련이 있습니다.

또한 각 한계가 설정된 경우 최소값 이하로 떨어지거나 최대값을 초과하는 곡선의 부분은 나머지 곡선 부분에 대해 색으로 오프셋될 수 있습니다.

## 첫 단계 - 모션

트레이스 기록을 제어할 수 있는 메뉴:

트레이스 창의 왼쪽 섹션을 마우스 오른쪽 버튼으로 클릭하면 상황에 맞는 메뉴가 나타납니다.

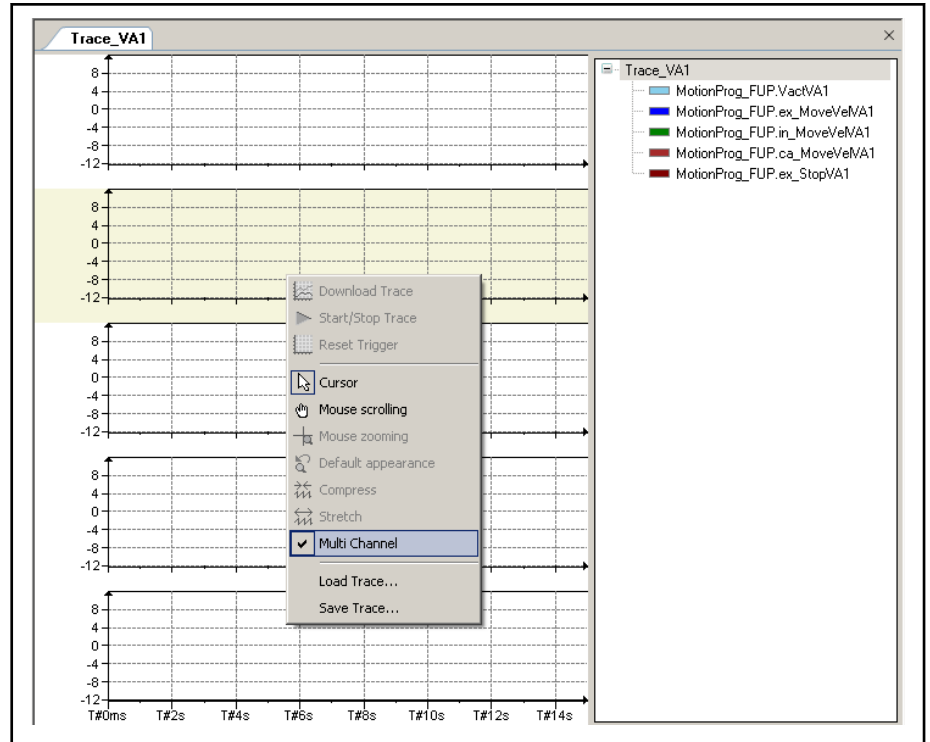


그림 5-40: 트레이스 기록을 제어할 수 있는 메뉴

**트레이스에 로그인(Login to Trace):** 실행 중인 응용 프로그램에 대해 컨트롤에 구성된 트레이스 레코드를 활성화하려면 먼저 "트레이스 응용 프로그램"을 명시적으로 로드해야 합니다.

**트레이스 시작/중지(Start/Stop Trace):** 이 명령은 트레이스 기록 표시를 시작하거나 중지합니다.

기록이 현재 중지된 경우 이 명령 앞에는 ► 아이콘이 표시됩니다.

기록이 현재 실행 중인 경우 이 명령 앞에는 ■ 아이콘이 표시되고, 이때 이 명령을 실행하면 기록이 중지됩니다. 즉, 트레이스 창에 현재 표시된 변수 값 곡선의 섹션이 "동결"됩니다.

**다중 채널(Multi Channel):** 이 명령을 사용하면 변경할 트레이스 곡선을 볼 수 있습니다. 기본적으로 모든 변수의 곡선이 동일한 차트에 표시됩니다. 다중 채널 보기에서는 각 변수의 곡선이 동일한 X 축을 사용하여 자체 다이어그램에서 시각화됩니다. 동시에 확대/축소하고 스크롤하면 모든 다이어그램의 X 축에 반영됩니다.

**커서(Cursor):** 이 명령이 활성화되어 있으면(기본값) 트레이스 창의 좌표계에서의 현재 커서 위치가 항상 프로그래밍 시스템의 상태 표시줄에 표시됩니다.

**마우스 스크롤(Mouse scrolling):** 이 모드에서는 커서가 양방향 화살표로 간단히 표시되고, 트레이스 창에서 커서를 이동하면 현재 표시된 기록의 시간 축을 왼쪽 또는 오른쪽으로 이동할 수 있습니다.

**기본 표시(Default appearance):** 이 명령을 사용하면 예를 들어, 확대/축소를 통해 기록이 변경되었더라도 기록 표시를 다시 기본값으로 재설정할 수 있습니다.

- 트레이스 수행:**
1. 응용 프로그램에 로그인  
컨트롤에 연결합니다.

## 첫 단계 - 모션

## 2. 트레이스에 로그인

응용 프로그램이 실행 중인 경우 "트레이스에 로그인"(Login to Trace) 상황에 맞는 메뉴를 통해 필요한 데이터를 컨트롤로 전송합니다.

그러면 트레이스가 자동으로 기록되기 시작합니다.

## 3. 값 쓰기

다음 값은 동일하게 아래와 같이 설정됩니다(입력 변수의 "값 쓰기"(Write Values) 102 페이지).

- ex\_MoveVelVA1: TRUE 로 설정
- ex\_StopVA1: TRUE 로 설정

## 4. 기록 끝내기

트레이스의 상황에 맞는 메뉴 항목 "트레이스 시작/중지"(Start/Stop Trace)를 통해 트레이스를 중지합니다.

## 5. 트레이스 기록 평가

기간 데이터를 설정합니다(아래 표 참조).

| 단축키                  | 트레이스 다이어그램에서의 동작                     |
|----------------------|--------------------------------------|
| <Alt>+<왼쪽/오른쪽> 화살표 키 | 모든 다이어그램에서 X 축을 가로로 스크롤              |
| <위쪽/아래쪽> 화살표 키       | 선택한 다이어그램만 가로로 스크롤                   |
| <+/-> 키              | 모든 다이어그램의 X 축 및 Y 축을 따라 좌표계 동시 확대/축소 |
| <Shift>+<+/-> 키      | X 축을 따라 모든 다이어그램을 가로로 스크롤            |
| <Ctrl>+<+/-> 키       | Y 축을 따라 모든 다이어그램을 세로로 스크롤            |
| <왼쪽/오른쪽> 화살표 키       | 커서를 왼쪽/오른쪽으로 스크롤                     |

표 5-1: 다중 채널이 활성화된 경우 트레이스 편집기의 단축키

## 첫 단계 - 모션

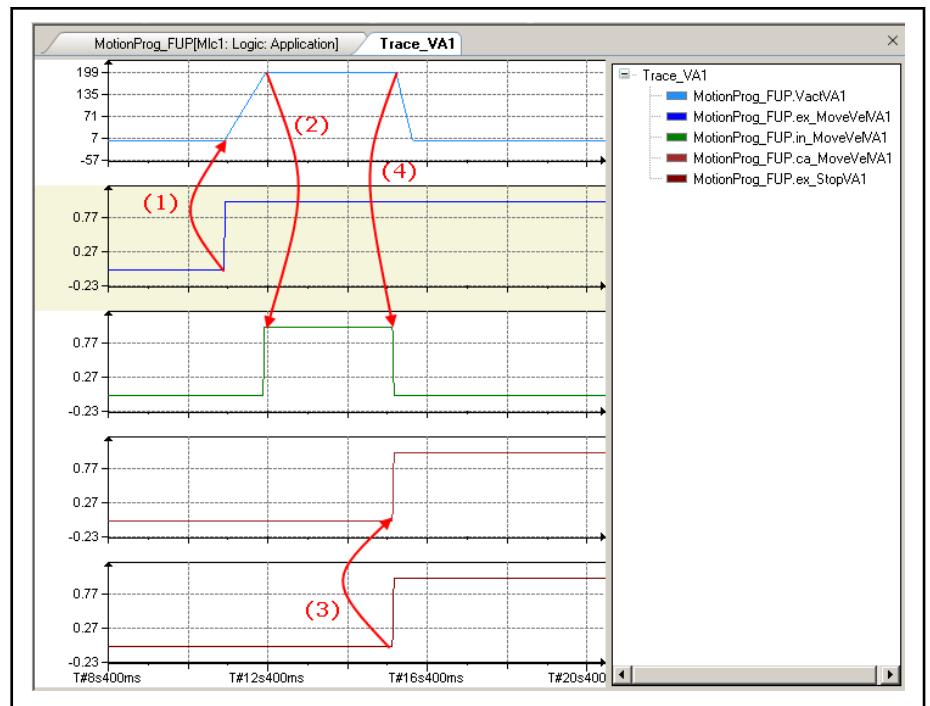


그림 5-41: 가상 축 VA1 에 대한 트레이스 기록

### 트레이스 기록 평가

- ① - 가상 축 시작
- ② - 설정 속도에 도달
- ③ - 가상 축 중지
- ④ - 가상 축의 속도가 줄어듦

가상 축의 동작은 동일한 매개변수를 입력할 경우 실제 축의 동작과 유사합니다. 가속도가  $20\text{rad/s}^2$  이고 감속도가  $50\text{rad/s}^2$  일 때의 속도 특성을 비교해 보십시오.

## 5.4.5 실제 축 RA1 프로그래밍(FBD(평선블럭다이어그램))



지금부터는 계속 "실제" 드라이브 기술이 필요합니다.

또는

대안으로 MC\_Power 없이 두 번째 가상 축을 선택합니다.

실제 축을 사용하려면 먼저 PLCopen 평선블럭 MC\_Power 의 인스턴스가 필요합니다.

그 후에는 MC\_GearIn 및 MC\_GearOut 의 인스턴스가 각각 하나씩 필요합니다.

MC\_Stop 의 인스턴스를 사용하여 실제 축을 중지합니다.

가상 축 VA1 프로그래밍, 프로그래밍 언어 ST(구조체언어) 섹션(117 페이지)과 유사하게 프로그래밍을 실행합니다.

### MC\_Power

새 네트워크를 삽입하려면 마우스 오른쪽 버튼으로 네트워크 3 을 클릭합니다.

#### 네트워크 삽입(아래)

MC\_Power 삽입 및 "PowerRA1" 인스턴스 선언

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

## 첫 단계 - 모션

"Enable" → BOOL 유형의 "en\_PowerRA1"

"Axis" → "AXIS\_REF" 유형의 "MlcVarGlobal.RA1"

출력:

"Status" → BOOL 유형의 "st\_PowerRA1"

"Error" → BOOL 유형의 "er\_PowerRA1"

"ErrorID" → ERROR\_CODE 유형의 "ID\_PowerRA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "Ident\_PowerRA1"

**MC\_GearIn**

새 네트워크를 삽입하려면 마우스 오른쪽 버튼으로 네트워크 4 를 클릭합니다.

**네트워크 삽입(아래)**

MC\_GearIn 삽입 및 "GearInRA1" 인스턴스 선언

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Execute" → BOOL 유형의 "ex\_GearInRA1"

"RatioNumerator" → 5

"RatioDenominator" → 10

"MasterFineadjust" → 20(%)

"Master" → "AXIS\_REF" 유형의 "MlcVarGlobal.VA1"

"Slave" → "AXIS\_REF" 유형의 "MlcVarGlobal.RA1"

출력:

"InGear" → BOOL 유형의 "ig\_GearInRA1"

"Active" → BOOL 유형의 "ac\_GearInRA1"

"CommandAborted" → BOOL 유형의 "ca\_GearInRA1"

"Error" → BOOL 유형의 "er\_GearInRA1"

"ErrorID" → ERROR\_CODE 유형의 "ID\_GearInRA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "Ident\_GearInRA1"

**MC\_GearOut**

새 네트워크를 삽입하려면 마우스 오른쪽 버튼으로 네트워크 5 를 클릭합니다.

**네트워크 삽입(아래)**

MC\_GearOut 삽입 및 "GearOutRA1" 인스턴스 선언

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Execute" → BOOL 유형의 "ex\_GearOutRA1"

"Slave" → "AXIS\_REF" 유형의 "MlcVarGlobal.RA1"

출력:

"Done" → BOOL 유형의 "do\_GearOutRA1"

"Active" → BOOL 유형의 "ac\_GearOutRA1"

"Error" → BOOL 유형의 "er\_GearOutRA1"

"ErrorID" → ERROR\_CODE 유형의 "ID\_GearOutRA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "Ident\_GearOutRA1"

## MC\_Stop

새 네트워크를 삽입하려면 마우스 오른쪽 버튼으로 네트워크 6 을 클릭합니다.

### 네트워크 삽입(아래)

MC\_Stop 삽입 및 "StopRA1" 인스턴스 선언

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Execute" → BOOL 유형의 "ex\_StopRA1"

"Deceleration" → 50(rad/s<sup>2</sup>)

"Axis" → "AXIS\_REF" 유형의 "MlcVarGlobal.RA1"

출력:

"Done" → BOOL 유형의 "do\_StopRA1"

"Active" → BOOL 유형의 "ac\_StopRA1"

"CommandAborted" → BOOL 유형의 "ca\_StopVA1"

"Error" → BOOL 유형의 "er\_StopRA1"

"ErrorID" → ERROR\_CODE 유형의 "ID\_StopRA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "Ident\_StopRA1"

## 실제 속도 "VactRA1"

마지막으로 축 RA1 "VactRA1"(데이터 유형 REAL)의 현재 속도를 읽을 수 있는 네트워크를 구현합니다.

현재 속도는 축의 데이터 집합(공유 메모리)에서 읽어 옵니다.

MLC 를 제어할 수 있는 모든 축의 데이터는 다음 배열에 있습니다.

**GVL\_Base.AxisData: ARRAY [1..xx<sup>2</sup>] OF ML\_AXISDATA\_SM**

여기에서 축 번호는 색인으로 사용됩니다.

변수의 이름을 다음과 같이 지정합니다.

**GVL\_Base.AxisData[1].rActualVelocity\_i,**

실제 축의 경우 축 번호는 1 입니다. (VA1 을 선언하기 전에 RA1 스캔이 실행 되었으므로 축 번호 1 은 RA1 입니다.)

- 새 네트워크를 삽입하려면 마우스 오른쪽 버튼으로 네트워크 7 을 클릭합니다.

### 네트워크 삽입(아래)

- 할당 삽입

- 오른쪽의 ??? 에 "VactRA1"을 삽입하고 <Enter> 키를 누른 다음 선언 창에서 "REAL" 유형을 선택하고 <Enter> 키를 누릅니다.

- 왼쪽의 ??? 으로 이동한 후  을 클릭하고 입력 보조툴을 사용하여 변수(Variables) ▶ ML\_BASE ▶ 전역변수리스트(Global Variable List) ▶ GVL\_Base ▶ AxisData 로 이동합니다.

2) 제어할 수 있는 축의 개수는 컨트롤 유형 및 해당 펌웨어에 따라 달라집니다(현재 L65 의 경우: 축 64 개).

## 첫 단계 - 모션

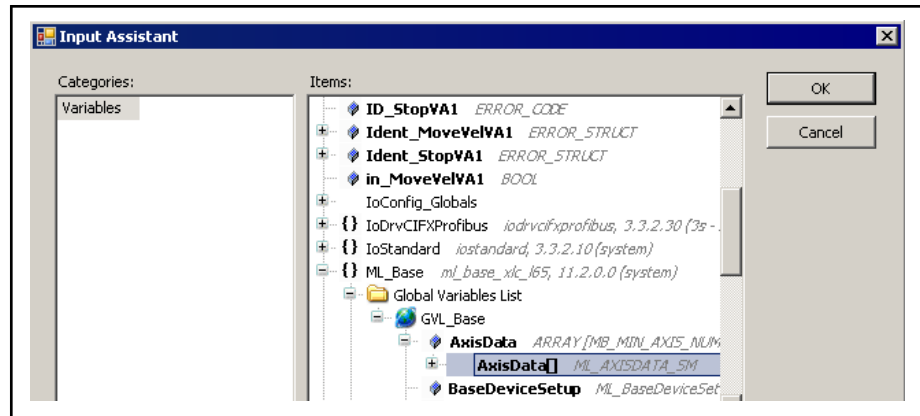


그림 5-42: ML\_Base.library 라이브러리의 전역변수리스트에 있는 실제 속도 값 VA1

Intellisense 기능을 통해 추가로 입력할 수 있습니다. 즉, "GVL\_Base.AxisData[1]."에 "."을 입력한 후 사용 가능한 모든 요소를 스크롤하여 선택할 수 있습니다.

<Enter> 키를 눌러 "rActualVelocity\_i"를 확인합니다.

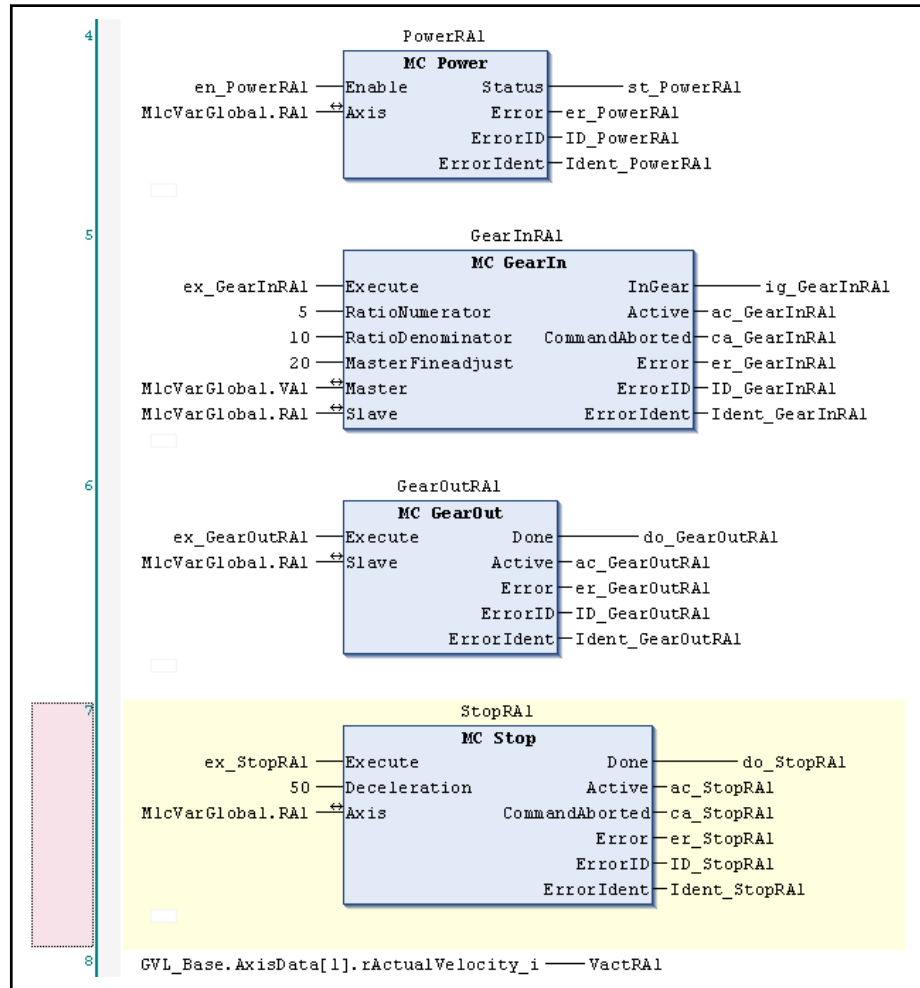


그림 5-43: 실제 축 구현

## 5.4.6 전체 PLC 프로젝트 다운로드(FBD(평선블럭다이어그램))

프로젝트는 PLC 하위 프로젝트 가상 축 VA1 다운로드(FBD(평선블럭다이어그램)) 섹션(100 페이지)에서 설명한 것과 동일하게 다운로드됩니다.

프롬프트가 표시된 후 새 프로젝트가 이전 프로젝트를 덮어씁니다.

## 5.4.7 워크플로 "가상 축 → 전자식 기어 → 실제 축"

다음 절차에서는 축과 기어를 시작하고 중지하는 방법을 보여 줍니다.

### 1. 실제 축의 전원 켜기

"값 쓰기"(Write Values)를 통해 "en\_PowerRA1"(MC\_Power)을 활성화합니다.

st\_PowerRA1= TRUE(MC\_Power)

### 2. 가상 축 시작

"값 쓰기"(Write Values)를 통해 "ex\_MoveVelVA1"(MC\_MoveVelocity)을 활성화합니다.

VactVA1 = 200rpm 에서 시작이 종료됩니다.

### 3. 이어서 실제 축 시작

"값 쓰기"(Write Values)를 통해 "ex\_GearInRA1"(MC\_GearIn)을 활성화합니다.

예상되는 기어 전송 속도는 다음과 같습니다.

$Tr = (RatioNumerator / RatioDenominator) * (1 + MasterFineadjust)$

VactVA1 = 200rpm 의 경우 VactRA1 = 120rpm 이 됩니다.



MasterFineadjust 값을 "%"로 지정합니다.

### 4. MC\_GearIn 이 "InGear"인 경우 "값 쓰기"(Write Values)를 통해 가상 축의 속도 "VcomVA1"을 변경할 수 있습니다. 변수를 두 번 클릭하고 원하는 값을 입력합니다. 실제 축이 이어서 시작됩니다.

### 5. "값 쓰기"(Write Values)를 통해 "ex\_GearInRA1"(MC\_GearIn)을 비활성화합니다.

in\_GearInRA1= FALSE, ac\_GearInRA1= TRUE

가상 축에 이어서 실제 축이 계속해서 시작됩니다.

### 6. 동시 발생 종료

"값 쓰기"(Write Values)를 통해 "ex\_GearOutRA1"(MC\_GearOut)을 활성화합니다.

가상 축에 대한 실제 축의 종속성이 일시 중단됩니다. 그러나 두 축이 계속해서 회전하는 속도는 바뀌지 않고 그대로 유지됩니다.

### 7. 설정 속도 "VcomVA1" 변경

변경된 설정 속도는 가상 축에만 영향을 미칩니다. "VactRA1"은 바뀌지 않고 그대로 유지됩니다.

### 8. "값 쓰기"(Write Values)를 통해 "ex\_GearOutRA1"을 비활성화합니다.

do\_GearOutRA1= FALSE

### 9. 가상 축 중지

"ex\_StopVA1"(MC\_Stop)을 활성화합니다.

가상 축이 VactVA1 = 0rpm 으로 감속됩니다.

### 10. "ex\_StopVA1"(MC\_Stop)을 비활성화합니다.

## 첫 단계 - 모션

## 11. 실제 축 중지

"ex\_StopRA1"(MC\_Stop)을 활성화합니다.

실제 축이 VactRA1= 0rpm 으로 감속됩니다. 중지된 후에는 이 위치를 유지합니다.

## 12. "ex\_StopRA1"(MC\_Stop)을 비활성화합니다.

실제 축의 활성 상태가 더 이상 유지되지 않습니다.

## 13. 실제 축의 전원 끄기

"값 쓰기"(Write Values)를 통해 "en\_PowerRA1"(MC\_Power)을 비활성화합니다.



RA1 에서 중지를 실행한 후 정지 모드에서만 가능합니다.

## 5.4.8 두 축의 모션 기록(트레이스), FBD(평선블럭다이어그램)

축 VA1 의 모션 기록(트레이스), FBD(평선블럭다이어그램) 섹션(104 페이지)에서 설명한 것과 동일한 방법으로 트레이스 도구를 사용하여 가상 축의 시작 및 동기화를 시각화했습니다.

다음 워크플로는 트레이스 기록을 위해 축과 기어를 시작하고 중지하는 방법을 보여 줍니다.

## 1. 실제 축의 전원 켜기

"en\_PowerRA1"(MC\_Power)을 활성화합니다.

## 2. 가상 축 시작

"ex\_MoveVelVA1"(MC\_MoveVelocity)을 활성화합니다.

## 3. 이어서 실제 축 시작

"ex\_GearInRA1"(MC\_GearIn)을 활성화합니다.

## 4. 가상 축의 속도 설정 값 증가

속도 설정 값을 증가시킵니다.

"ex\_MoveVelVA1" TRUE ⇒ FALSE ⇒ TRUE 를 통해 켭니다.

다음 그림은 트레이스 결과를 보여 줍니다.

## 첫 단계 - 모션

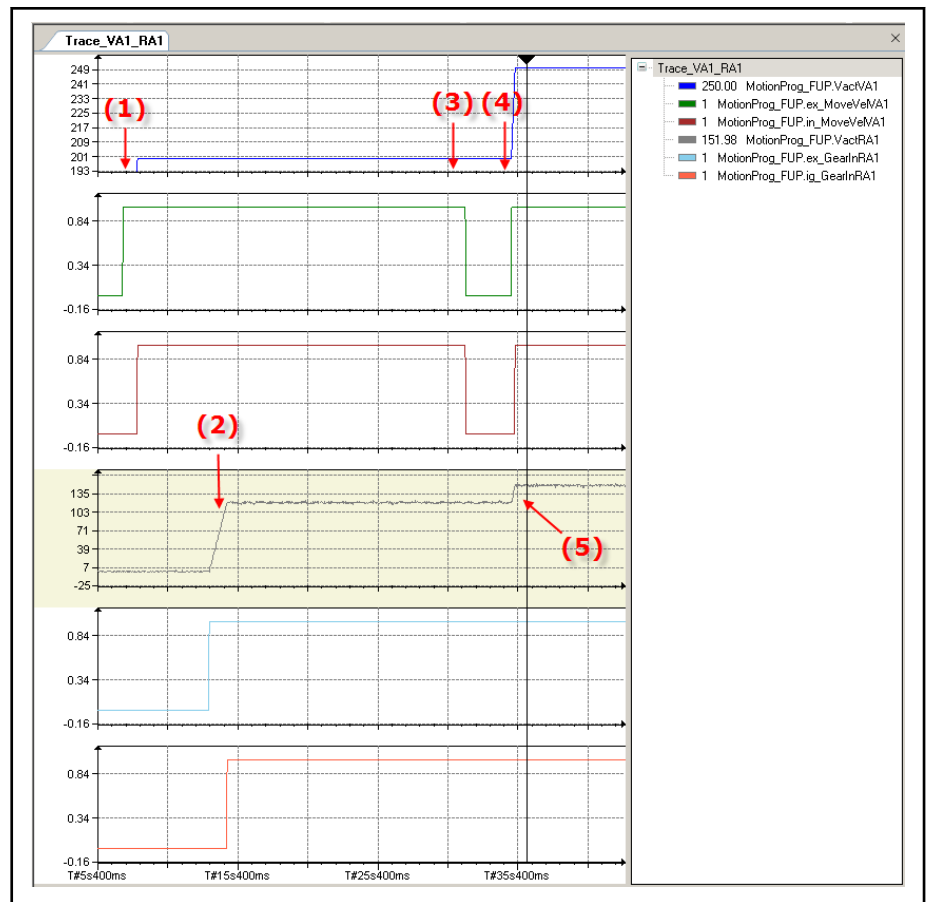


그림 5-44: 두 축의 트레이스 기록

### 트레이스 기록 평가

- ① - 가상 축 시작
- ② - 기어 기능 활성화, 실제 축 시작
- ③ - 가상 축의 설정 속도 증가, "ex\_MoveVelVA1" TRUE ⇒ FALSE
- ③ - 가상 축의 설정 속도 활성화, "ex\_MoveVelVA1" FALSE ⇒ TRUE
- ⑤ - 실제 축에 대한 반응 증가

## 5.5 ST(구조체언어)에서 모션 스퀀스 프로그래밍

### 5.5.1 ST(구조체언어)에서 모션 스퀀스 프로그래밍, 일반 정보

PLC 프로그램에서 PLCOpen 평선블럭(EN 61131-3 에 따라 프로그래밍 시스템 IndraLogic 2G)을 사용하여 모션 스퀀스를 프로그래밍합니다.

DDC\_Control 컨트롤, 실제 축 RA1 및 가상 축 VA1 을 사용하여 프로젝트를 만들 때 모션 스퀀스를 프로그래밍하는 첫 번째 작업은 "로직"(Logic) 폴더 아래에서 시작되었습니다.

## 첫 단계 - 모션

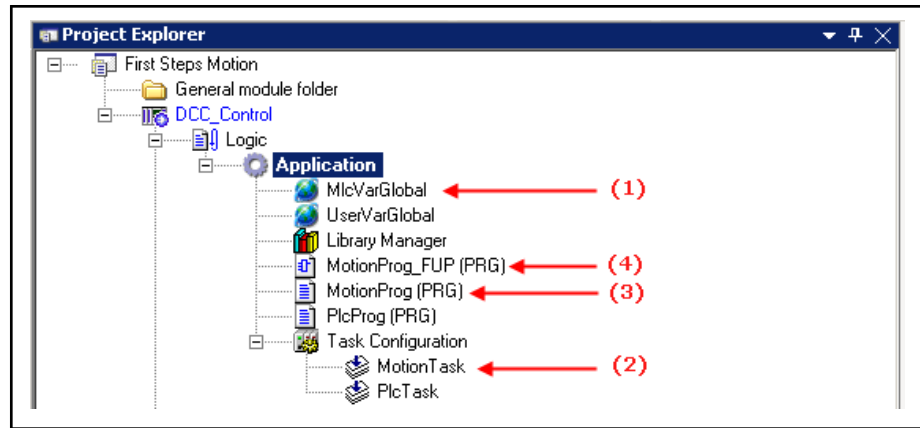


그림 5-45: "로직"(Logic) 노드를 확장한 후의 프로젝트 탐색기

다음 프로그램이 생성되었습니다.

- **MotionProg (3)**(ST(구조체언어) 사전 선택)
- **PlcProg**(ST(구조체언어) 사전 선택), 사용 안 함

세 번째 프로그램 **MotionProg\_FBD ( 4 )**는 응용 프로그램(Application)을 마우스 오른쪽 버튼으로 클릭한 후 추가(Add) ▶ POU(프로그램 유닛)에서 이름을 입력하고 프로그래밍 언어로 FBD(평선블록 다이어그램)를 선택하여 수동으로 추가했습니다.

프로그램은 현재 비어 있습니다.

"작업 구성"(Task Configuration) 아래에 다음 작업이 생성되었습니다.

- **MotionTask ( 2 )**, "MotionProg"의 경우 (우선 순위 2, sercos 작업에 의해 외부에서 이벤트 트리거됨, 위치독 활성)  
이 작업 아래에서 실행 중인 프로그램은 sercos 주기의 지정된 파트 내에 완전히 처리해야 합니다. 이러한 프로그램이 이 파트 내에 없으면 컨트롤이 중지됩니다.
- **PlcTask**, PlcProg 의 경우 (낮은 우선 순위 주기 작업), 사용 안 함  
남은 시간에 프로그램을 조금씩 처리합니다. 즉, 필요한 경우 여러 주기 동안 처리됩니다.

다음에 대해 전역 변수/전역 상수가 생성되었습니다.

- **UserVarGlobal**: 사용자에 대해 예약됨, "첫 단계 - 모션"(First Steps - Motion)에서 사용 안 함
- **MlcVarGlobal ( 1 )**: RA1 및 VA1 축에 대한 주소 상수 포함

#### 프로그램

```
(* Automated generated code by MLC. *)
(* Please don't edit, the code will be overwritten...*)
VAR_GLOBAL CONSTANT
(*virtual axis "VA1" with axis number 2*)
VA1: AXIS_REF :=(CntrlNo:=LOCAL_CNTRL,AxisNo:=AXIS_2);
(*real axis "RA1" with axis number 1*)
RA1: AXIS_REF :=(CntrlNo:=LOCAL_CNTRL,AxisNo:=AXIS_1);
(*...*)
END_VAR
```



"첫 단계 - 모션"(First Steps - Motion)에는 MotionTask 를 사용합니다.

필요한 프로그래밍 언어에 따라 **MotionProg** 또는 **MotionProg\_FBD** 가 할당됩니다.

## 첫 단계 - 모션

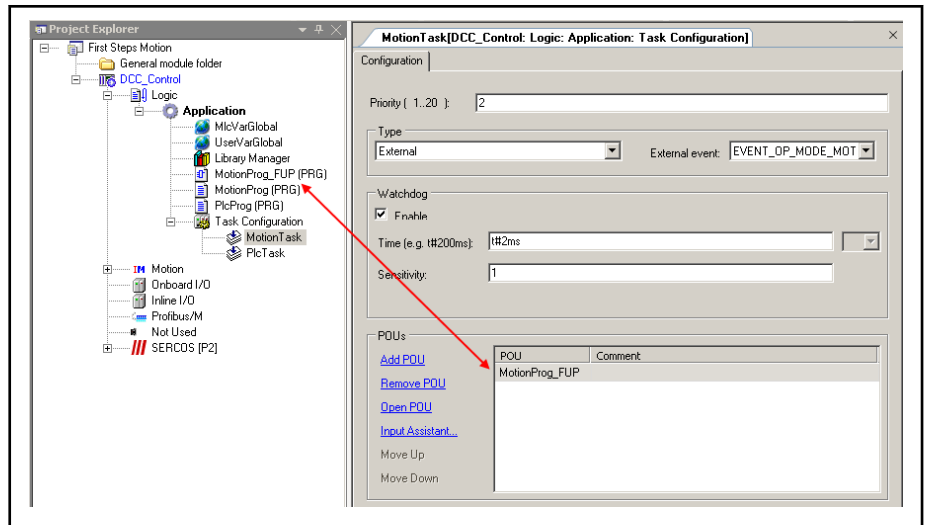


그림 5-46: 호출 삭제: MotionProg\_FBD, 호출 추가: MotionProg

### 5.5.2 가상 축 VA1 프로그래밍, 프로그래밍 언어 ST(구조체언어)



이 섹션에서는 프로그래밍 언어로 ST(구조체언어)를 사용합니다. 기능 면에서 볼 때 프로그래밍 언어 FBD(평선블럭다이어그램)를 사용한 섹션과 동일한 결과가 발생합니다.

프로젝트 목표 첫 단계 - 모션, 81 페이지와 마찬가지로 PLCopen 블럭 MC\_MoveVelocity 및 MC\_Stop 의 인스턴스는 "VA1"을 사용하여 호출합니다.



가상 축에는 MC\_Power 인스턴스가 필요하지 않습니다.

아래에 설명된 대로 진행합니다.

1. "MotionProg" 프로그램에 MotionTask 를 할당합니다.
2. "MotionProg" 프로그램을 두 번 클릭하여 엽니다.

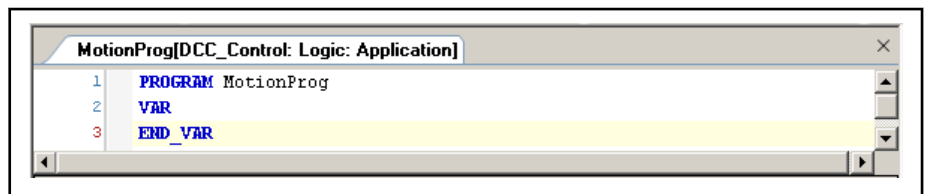


그림 5-47: 더미 줄 삭제 후 ST(구조체언어)의 새 프로그램

FBD(평선블럭다이어그램)로 프로그래밍할 때와 달리 ST(구조체언어)에서는 먼저 선언영역 VAR ... END\_VAR 에 평선블럭의 인스턴스를 선언해야 합니다.

그런 다음에만 구현에서 이러한 인스턴스를 사용할 수 있습니다.

먼저 VAR ... END\_VAR 섹션에서 평선블럭 MC\_MoveVelocity 의 인스턴스 이름 "MoveVelVA1"을 입력합니다. 해당 이름을 마우스 오른쪽 버튼으로 클릭하고 상황에 맞는 메뉴에서 입력 보조를 선택하거나 <F2> 키를 누릅니다.

## 첫 단계 - 모션

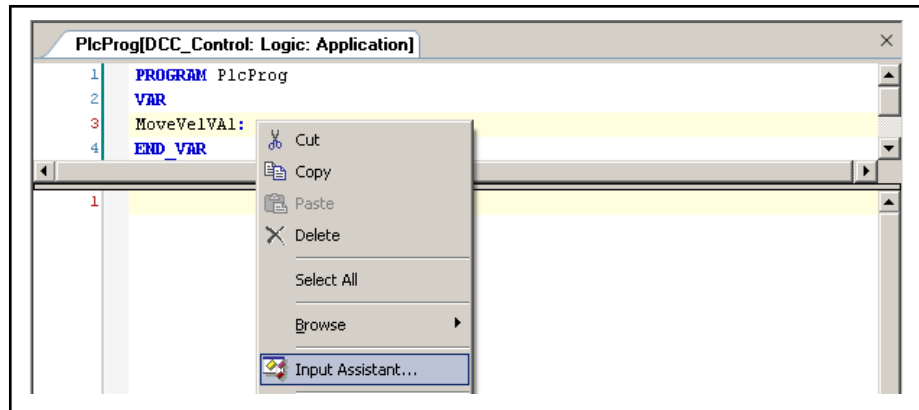


그림 5-48: MC\_MoveVelocity 유형의 인스턴스 MoveVelVA1 선언

이렇게 하려면 열린 선택 창에서 다음 항목을 선택합니다.

사용자 정의 유형(User-defined Types) ▶ ML\_PLCOPEN ▶ POU(프로그램 유닛) ▶ 모션(Motion) ▶ MC\_MoveVelocity

"인수와 함께 삽입"(Insert with arguments) 옵션을 선택합니다.

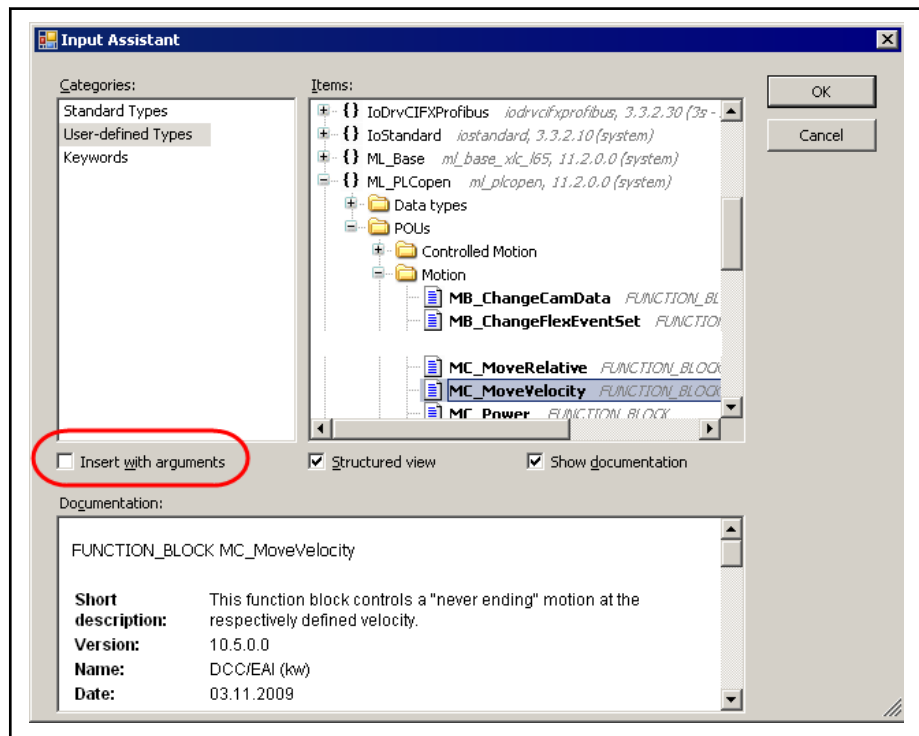


그림 5-49: 선택 창 "평선블럭"

"확인"(OK)을 클릭하여 확인한 후 줄 마지막에 ";"를 입력하여 완료합니다.

편집기의 구현 섹션에서 첫째 줄로 이동한 후 <F2> 키를 클릭하여 입력 보조 툴을 표시합니다.

열린 창에서 인스턴스 호출(Instance Calls) ▶ MoveVelVA1 을 선택합니다.

"인수와 함께 삽입"(Insert with arguments) 옵션을 선택합니다.

첫 단계 - 모션

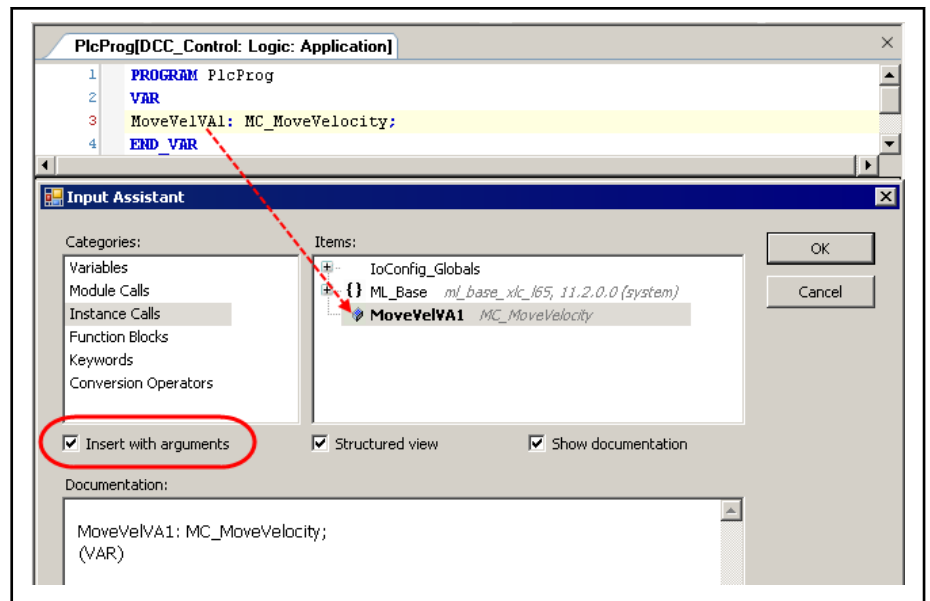


그림 5-50: 인스턴스를 구현으로 전송

"확인"(OK)을 클릭하여 인스턴스의 삽입을 확인합니다.

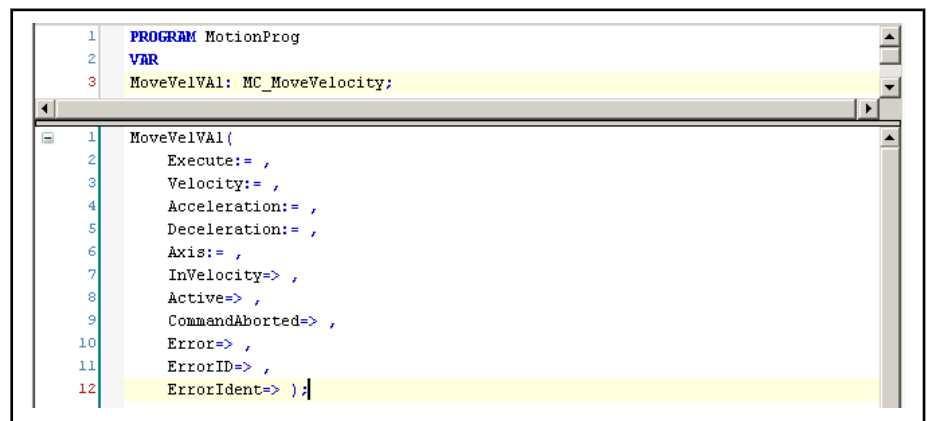


그림 5-51: 평선블럭 인스턴스의 전반적인 디자인

FB(평선블럭) 입력을 할당합니다.

"Execute" 입력부터 시작합니다.

- 이름 "ex\_MoveVelVA1"을 입력하고 "Enter" 키를 눌러 완료합니다.
- "Enter" 키를 누르면 바로 열리는 "변수 선언" 창을 확인합니다.

"Velocity" 입력을 할당하려면 다음 단계를 진행합니다.

- 이름 "VcomVA1"을 입력하고 "Enter" 키를 눌러 확인합니다.
- 유형으로 "REAL"을 입력하고 초기 값으로 200(rpm)을 입력한 후 "Enter" 키를 눌러 확인합니다.

"Acceleration" 입력으로 20(rad/s²)을 입력하고 "Deceleration"에 20(rad/s²)을 입력합니다.



평선블럭 입력에 바로 숫자를 입력합니다.

또는 개별 값을 갖는 변수를 사용할 수 있습니다.

"Axis" 입력을 할당하려면 다음 단계를 진행합니다.

- 축 입력을 마우스 오른쪽 버튼을 클릭하여 근처에 표시되는 입력 보조를 선택합니다.

## 첫 단계 - 모션

변수(Variables) ▶ 응용 프로그램(Application) ▶ MlcVarGlobal ▶ VA1 을 선택합니다.

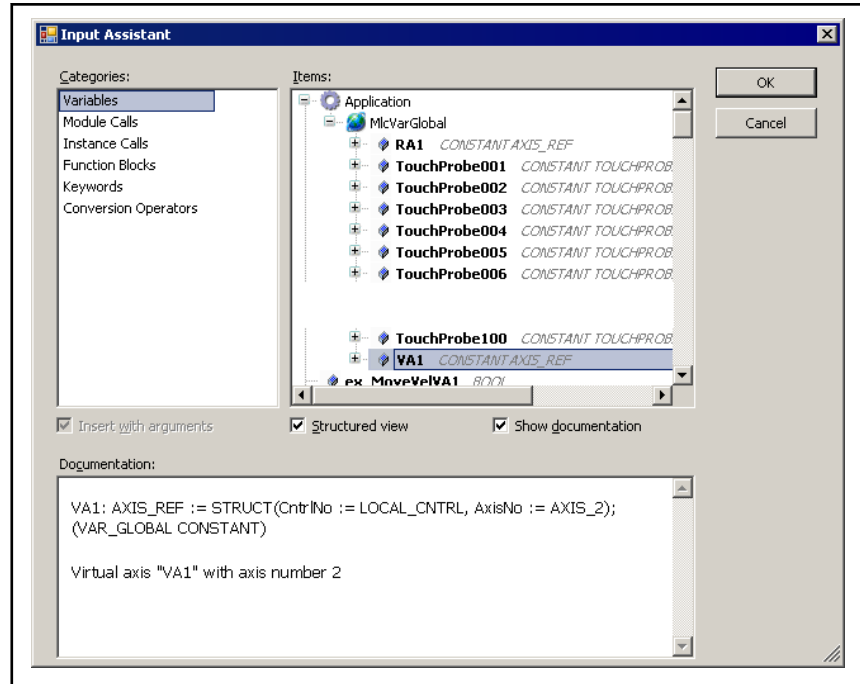


그림 5-52: "MC\_MoveVelocity" 유형의 평선블럭을 할당할 때 축 인스턴스 선택

- "확인"(OK)을 클릭하여 확인합니다.



전역 상수의 이름은 다음과 같이 폴더 이름과 요소 이름으로 구성됩니다.

MlcVarGlobal.VA1

평선블럭이 제대로 작동하는지 여부에 대한 피드백을 받으려면 해당 출력을 할당합니다.

- BOOL 유형의 "iv\_MoveVelVA1" 변수를 사용하여 FB(평선블럭)의 "InVelocity" 출력을 할당합니다.
- 이름 "iv\_MoveVelVA1"을 입력한 다음 <Enter> 키를 누릅니다.
- 열려 있는 "변수 선언" 창은 <Enter> 키를 눌러 바로 확인할 수 있습니다.
- "CommandAborted" 출력에 "ca\_MoveVelVA1"을 할당합니다.
- "Error" 출력에 "er\_MoveVelVA1"을 할당합니다.
- "ErrorID" 출력에 "ID\_MoveVelVA1"을 할당합니다.
- "ErrorIdent" 출력에 "Ident\_MoveVelVA1"을 할당합니다.

MC\_MoveVelocity 유형의 PLCopen FB 가 해당 작업에 따라 할당됩니다. 선언 영역이 자동으로 완성되었습니다.

첫 단계 - 모션

```

2  VAR
3  MoveVelVal: MC_MoveVelocity;
4  ex_MoveVelVal: BOOL;
5  VcomVal: REAL := 200;
6  iv_MoveVelVal: BOOL;
7  ac_MoveVelVal: BOOL;
8  ca_MoveVelVal: BOOL;
9  er_MoveVelVal: BOOL;
10 ID_MoveVelVal: ERROR_CODE;
11 Ident_MoveVelVal: ERROR_STRUCT;
12 END VAR

1  MoveVelVal(
2  Execute:= ex_MoveVelVal,
3  Velocity:= VcomVal,
4  Acceleration:= 20,
5  Deceleration:= 20,
6  Axis:= MlcVarGlobal.VA1,
7  InVelocity=> iv_MoveVelVal,
8  Active=> ac_MoveVelVal,
9  CommandAborted=> ca_MoveVelVal,
10 Error=> er_MoveVelVal,
11 ErrorID=> ID_MoveVelVal,
12 ErrorIdent=> Ident_MoveVelVal);

```

그림 5-53: "MC\_MoveVelocity" 유형의 PLCopen FB 가 삽입 및 할당됨



평선블럭이 완료될 때마다 <Shift>+<F11>을 사용하여 테스트 컴파일을 실행합니다. 오류가 있으면 바로 지우십시오.

PLCopen FB MC\_Stop 의 인스턴스를 생성합니다.

**MC\_Stop**

MC\_Stop 의 인스턴스 "StopVA1"을 선언하고 삽입합니다.

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Execute" → BOOL 유형의 "ex\_StopVA1"

"Deceleration" → 50

"Axis" → "AXIS\_REF" 유형의 "MlcVarGlobal.VA1"

출력:

"Done" → BOOL 유형의 "do\_StopVA1"

"Active" → BOOL 유형의 "ac\_StopVA1"

"CommandAborted" → BOOL 유형의 "ca\_StopVA1"

"Error" → BOOL 유형의 "er\_StopVA1"

"ErrorID" → ERROR\_CODE 유형의 "ID\_StopVA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "Ident\_StopVA1"

**실제 속도 "VactVA1"**

마지막으로 축 VA1 "VactVA1"(데이터 유형 REAL)의 현재 속도를 읽을 수 있는 네트워크를 구현합니다.

현재 속도는 축의 데이터 집합(공유 메모리)에서 읽어 옵니다.

컨트롤을 제어할 수 있는 모든 축의 데이터는 다음 배열에 있습니다.

**GVL\_Base.AxisData: ARRAY [1..xx<sup>3)</sup>] OF ML\_AXISDATA\_SM**

여기에서 축 번호는 색인으로 사용됩니다.

변수의 이름을 다음과 같이 지정합니다.

3) 제어할 수 있는 축의 개수는 컨트롤 유형 및 해당 펌웨어에 따라 달라집니다(현재 L65의 경우: 축 64 개).

## 첫 단계 - 모션

**GVL\_Base.AxisData[2].rActualVelocity\_i,**

가상 축의 경우 축 번호는 2 입니다. VA1 을 선언하기 전에 RA1 스캔이 실행되었으므로 축 번호 1 은 RA1 입니다.

- 새 줄을 삽입합니다.
- VactVA1:= 을 삽입하고 선언을 완료합니다.
- 입력 보조툴을 사용하여 변수(Variables) ▶ ML\_BASE ▶ 전역변수리스트 (Global Variable List) ▶ GVL\_Base ▶ AxisData 로 이동합니다.

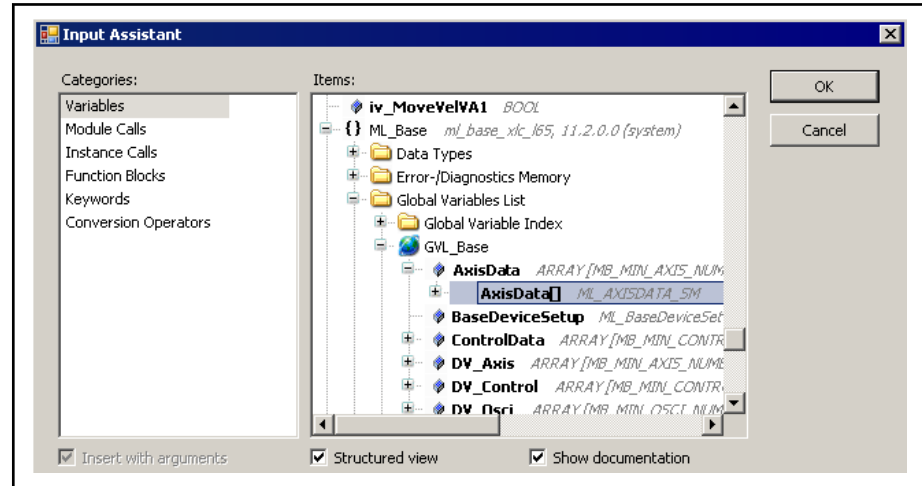


그림 5-54: ML\_Base.library 라이브러리의 전역변수리스트에 있는 실제 속도 값 VA1

Intellisense 기능을 통해 추가로 입력할 수 있습니다. 즉, "GVL\_Base.AxisData[2]."에 대해 "."을 입력한 후 사용 가능한 모든 요소를 열어 선택할 수 있습니다.

<Enter> 키를 눌러 "rActualVelocity\_i"를 확인합니다.

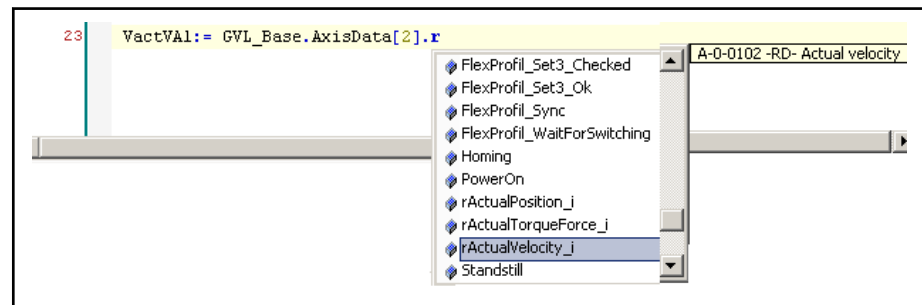


그림 5-55: "ActualVelocity" 변수 선택

- 줄 마지막에는 ";"을 입력해야 합니다.

작업 할당과 관련된 모션 스크스의 구현이 다음과 같이 완료되었습니다.

```

1  PROGRAM MotionProg
2  VAR
3  MoveVelVal: MC_MoveVelocity;
4  ex_MoveVelVal: BOOL;
5  VcomVal: REAL := 200;
6  iv_MoveVelVal: BOOL;
7  ac_MoveVelVal: BOOL;
8
9  MoveVelVal(
10     Execute:= ex_MoveVelVal,
11     Velocity:= VcomVal,
12     Acceleration:= 20,
13     Deceleration:= 20,
14     Axis:= MlcVarGlobal.Val,
15     InVelocity:= iv_MoveVelVal,
16     Active:= ac_MoveVelVal,
17     CommandAborted:= ca_MoveVelVal,
18     Error:= er_MoveVelVal,
19     ErrorID:= ID_MoveVelVal,
20     ErrorIdent:= Ident_MoveVelVal);
21
22 StopVal(
23     Execute:= ex_StopVal,
24     Deceleration:= 50,
25     Axis:= MlcVarGlobal.Val,
26     Done:= do_StopVal,
27     Active:= ac_StopVal,
28     CommandAborted:= ca_StopVal,
29     Error:= er_StopVal,
30     ErrorID:= ID_StopVal,
31     ErrorIdent:= Ident_StopVal);
32
33 VactVal:= GVL_Base.AxisData[2].rActualVelocity_i;

```

그림 5-56: 가상 축 VA1 구현

### 5.5.3 PLC 하위 프로젝트 가상 축 VA1 다운로드(ST(구조체언어))

위에서 구현된 모션 스퀀스는 컴파일하고 컨트롤에 적용한 후 시작해야 합니다.

다음을 통해 컴파일에 액세스할 수 있습니다.

- **만들기(Create) ▶ 작성(Build),**  
응용 프로그램의 추가된 부분 컴파일
- **만들기(Create) ▶ 다시 작성(Rebuild),**  
전체 응용 프로그램 컴파일

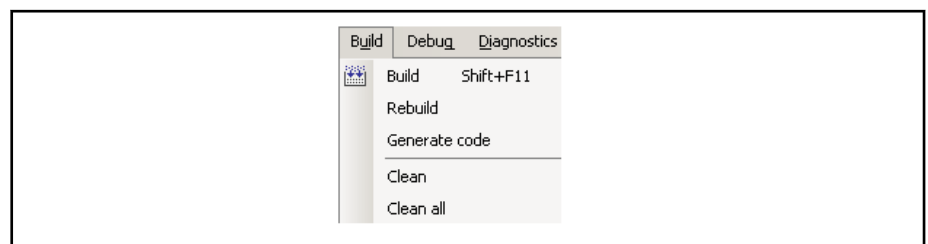


그림 5-57: 응용 프로그램 작성

컴파일된 데이터를 대상 컨트롤에 적용하려면 메뉴 항목 디버그(Debug) ▶ 로그인(Login)을 선택합니다. 아래 그림을 참조하십시오.

## 첫 단계 - 모션

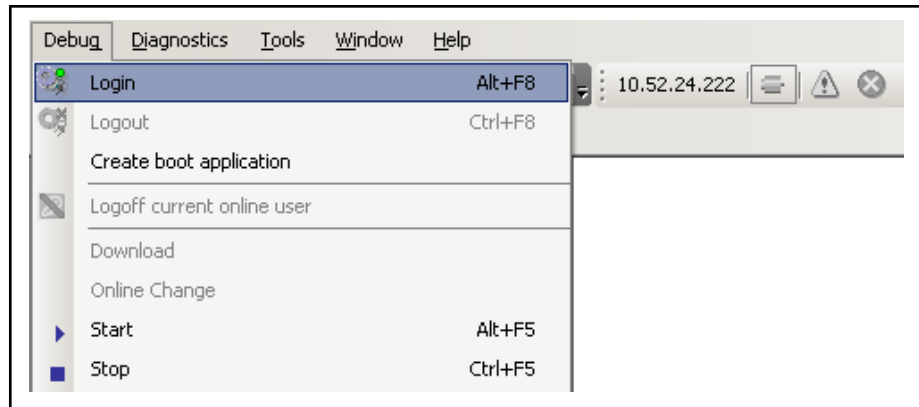


그림 5-58: 컴파일된 데이터를 컨트롤로 전송

응용 프로그램이 이미 로드되었는지 그리고 이 응용 프로그램이 현재 응용 프로그램의 이전 버전인지에 따라 다운로드 동의 여부를 묻는 메시지가 나타납니다.

응용 프로그램은 성공적인 전송 후에도 계속 중지되어 있으며(프로젝트 탐색기 및 상태 표시줄에 빨간색으로 표시됨) 값을 표시하도록 선언 및 구현에 필요한 변수가 준비됩니다. 그러나 값이 유효하지 않습니다.

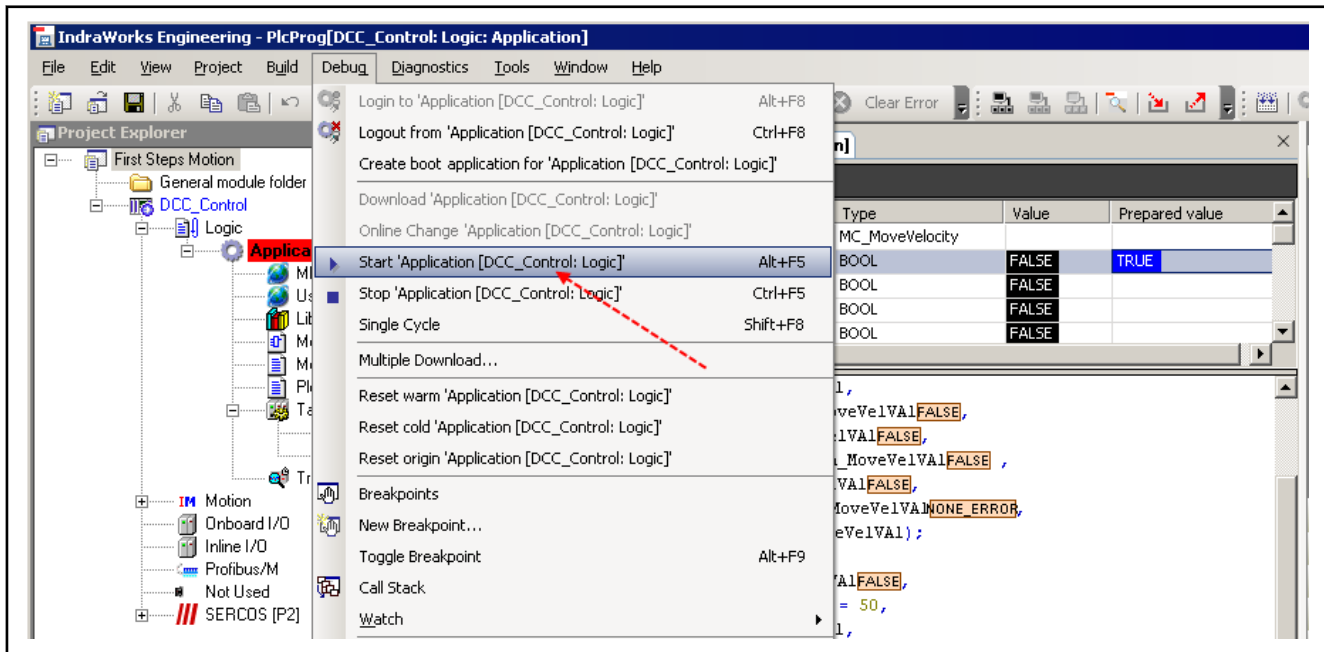


그림 5-59: 컨트롤에서 프로그램 처리 시작(1)

디버그(Debug) ▶ 시작(Start)을 통해 컨트롤에서 처리를 시작합니다.

첫 단계 - 모션

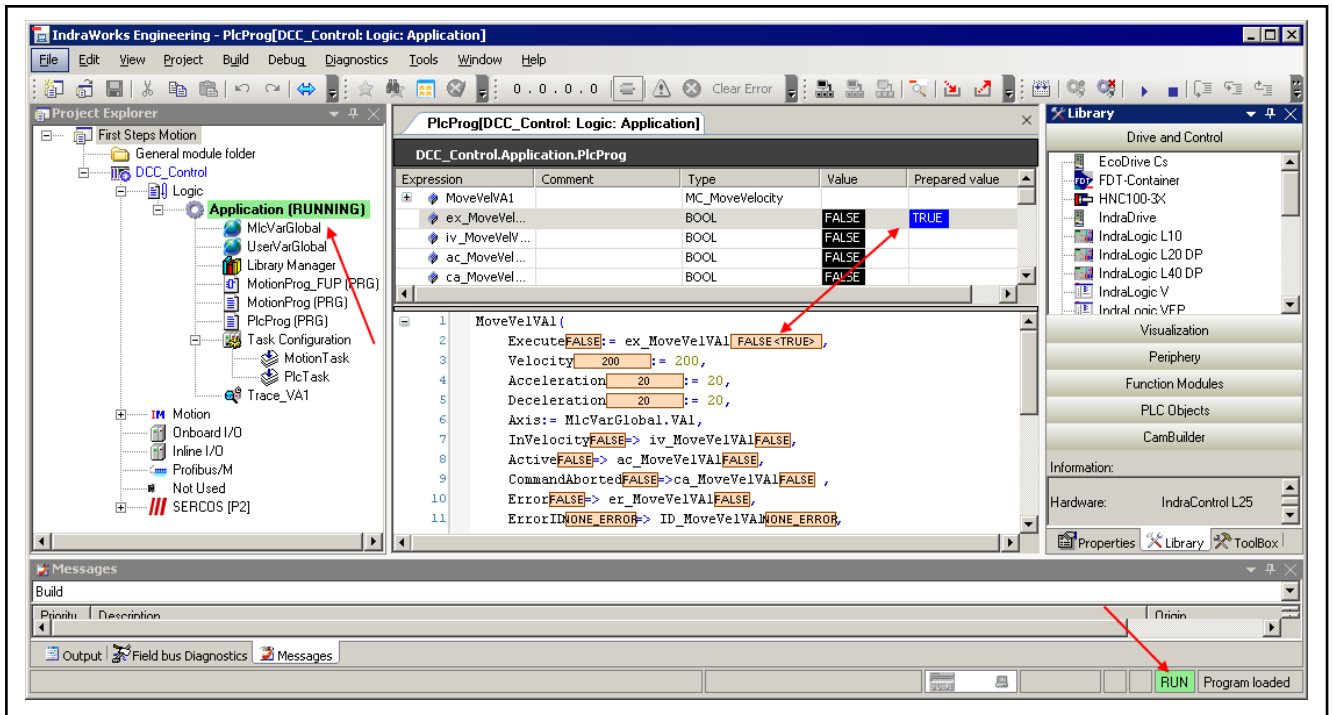


그림 5-60: 컨트롤에서 프로그램 처리 시작(2)

프로그래밍된 평선블럭의 처리가 차례로 활성화되면 모션 프로세스가 자동으로 시작됩니다.

"값 쓰기", 입력 변수

"MC\_MoveVelocity" 유형의 "MoveVelVA1"을 처리하려면 FB(평선블럭)의 "Execute" 입력 시 상승 에지가 발생해야 합니다.

마우스 왼쪽 버튼을 사용하여 입력 변수 "ex\_MoveVelVA1" 뒤에 있는 "FALSE"를 두 번 클릭합니다. "값 준비"(Prepare Value) 창이 열립니다. "TRUE"를 입력하고 확인합니다.

이제 "ex\_MoveVelVA1" 뒤의 "FALSE<TRUE>"가 강조되고 선언 줄에 "TRUE"가 나타납니다.

## 첫 단계 - 모션

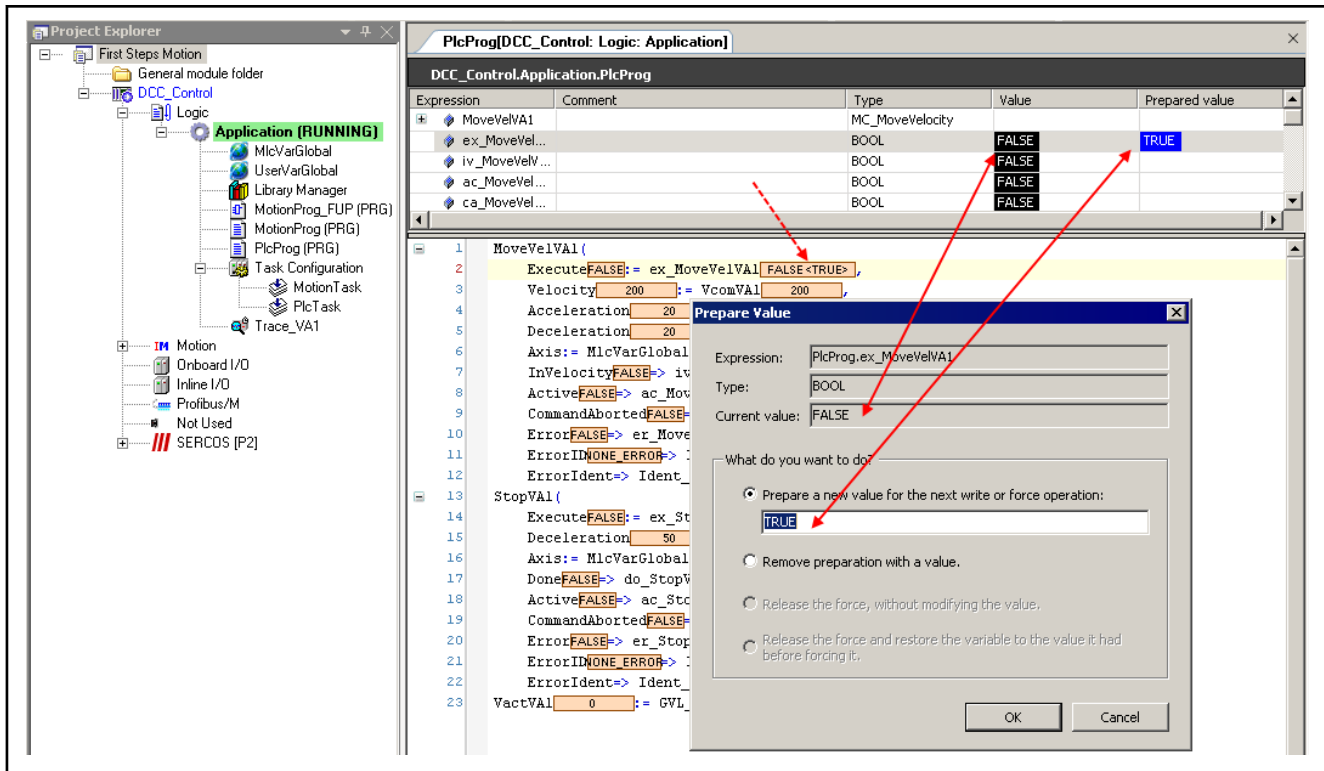


그림 5-61: "값 쓰기"(Write values) 준비

디버그(Debug) ▶ 값 쓰기(Write Values)(**Ctrl+F7**)를 통해 컨트롤의 "ex\_MoveVelVA1"을 "TRUE"로 설정합니다.



- "값 쓰기"(Write Values)는 다음 주기가 시작할 때 변수를 원하는 값으로 한 번 설정하는 것을 의미합니다.
- "값 강제 적용"(Force Values)은 다음 주기가 시작할 때 변수를 원하는 값으로 영구적으로 설정하는 것을 의미합니다.

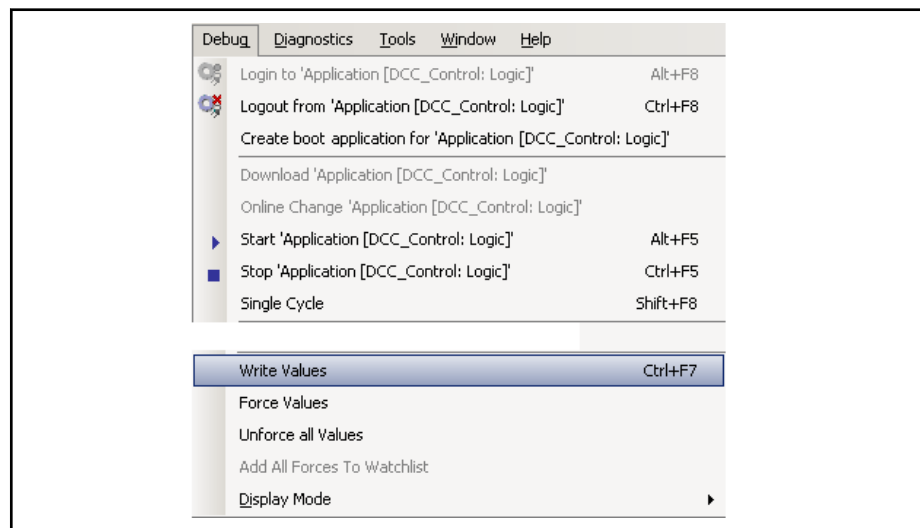


그림 5-62: 값 쓰기(Write Values)

생성된 상승 예지는 "MoveVelVA1" 평선블럭의 "InVelocity" 및 "Active" 출력을 "TRUE"(1)로 설정합니다. 아래 그림을 참조하십시오.

속은 200rpm(2)으로 이동합니다.

첫 단계 - 모션

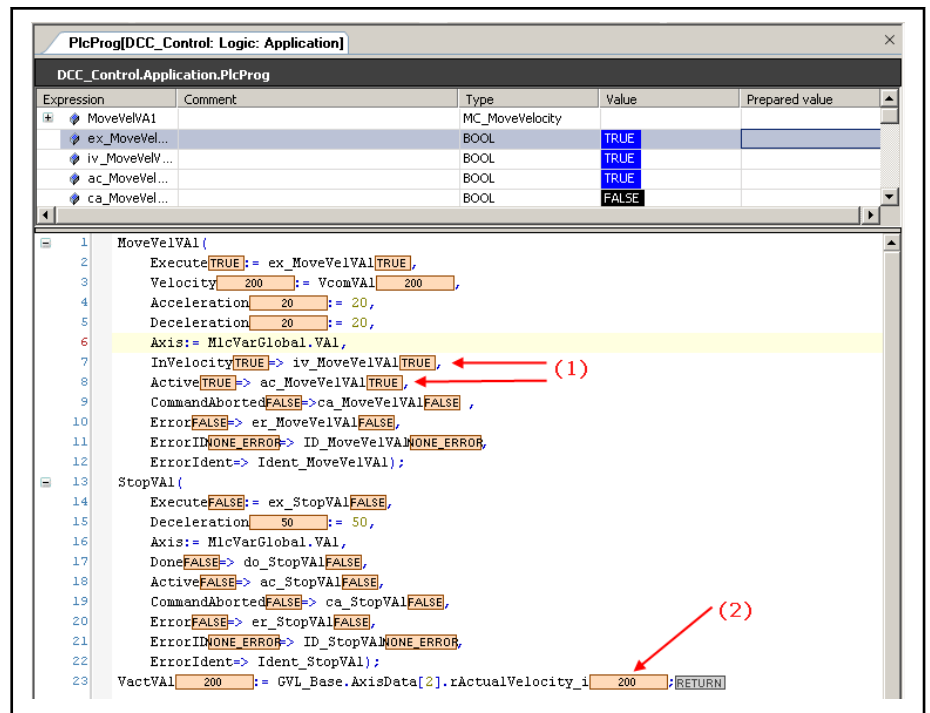


그림 5-63: "Execute" 입력의 "값 쓰기"(Write Values) 후

"ex\_MoveVelVA1"을 다시 "FALSE"로 전환합니다. 축은 계속해서 200rpm②으로 이동합니다.

MC\_Stop 유형의 "StopVA1" 평선블럭의 "Execute" 입력에서 "x\_StopVA1" 변수를 설정하여 축 모션을 중지할 수 있습니다.

PLC 를 중지하고 로그아웃합니다.

## 5.5.4 축 VA1 의 모션 기록(트레이스), ST(구조체언어)

IndraLogic 은 모션 프로세스를 트레이스할 수 있는 트레이스 도구를 제공합니다. 트레이스 도구를 사용하여 다음 항목을 기록할 수 있습니다.

- 축 속도 "VactVA1"
- "MoveVelVA1"(MC\_MoveVelocity)의 경우
  - "ex\_MoveVelVA1"(Execute)
  - "in\_MoveVelVA1"(InVelocity)
  - "ca\_MoveVelVA1"(CommandAborted)

및

- "StopVA1"(MC\_Stop)의 경우
  - "ex\_StopVA1"(Execute)

이렇게 하려면 먼저 트레이스 기록을 구성해야 합니다.

**응용 프로그램(Application)의 상황에 맞는 메뉴 추가(Add) ▶ 트레이스...(Trace...)**을 통해 "Trace\_VA1" 개체를 삽입합니다. 개체를 두 번 클릭하여 엽니다.

## 첫 단계 - 모션

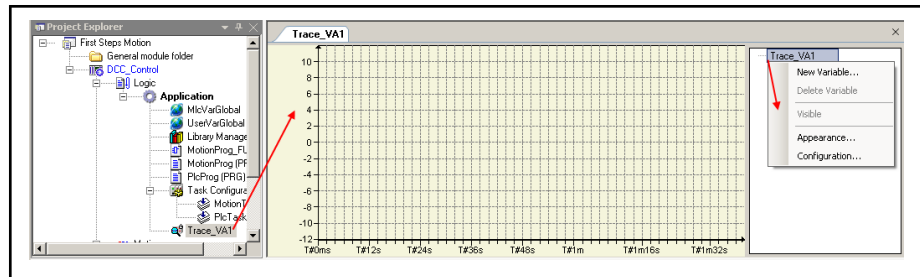


그림 5-64: 대화 상자: 상황에 맞는 메뉴가 표시된 새 트레이스 객체 "Trace\_VA1" 페이지의 오른쪽에서 상황에 맞는 메뉴를 통해 트레이스 객체와 관련하여 다음 작업을 수행할 수 있습니다.

- 구성...(Configuration...)
- 표시...(Appearance...)
- 새 변수...(New Variable...)

구성...(Configuration...) 먼저, 다음과 같이 새 트레이스 객체를 구성해야 합니다.

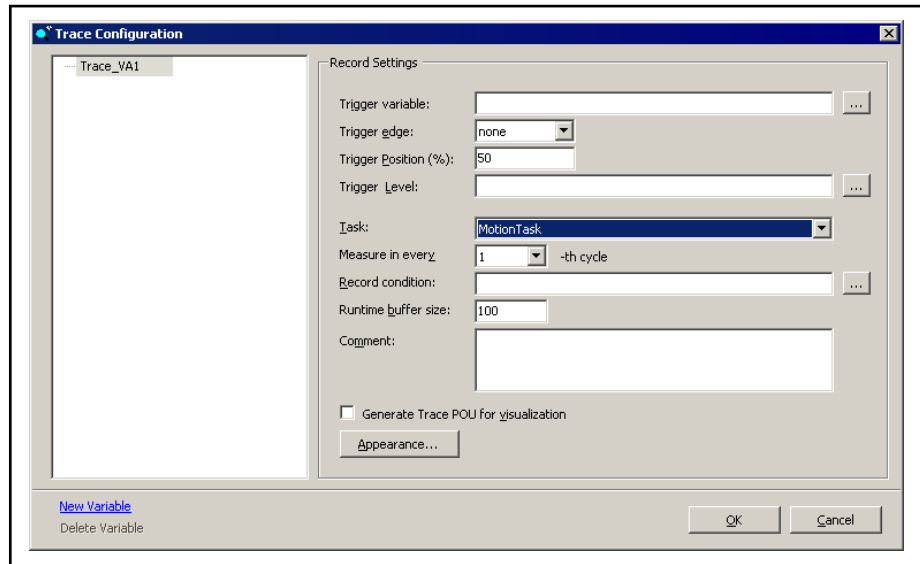


그림 5-65: 대화 상자: 새 트레이스 객체 "Trace\_VA1", 최소 구성

## 필요한 설정

- 작업: **MotionTask**
- 20 번째 주기마다 기록 즉,  $20 \times 2\text{ms} = 40\text{ms}$ -

**작업:(Task:)** 사용 가능한 작업 목록에서 처리한 후에 트레이스 변수의 값을 읽을 작업을 선택합니다. 기본값: 작업 구성 트리의 첫 번째 작업

**다음 주기마다 측정 ... 번째 주기:(Measure in every ... -th cycle:)** 트레이스 변수의 값을 다시 읽기 전에 경과할 주기 횟수를 입력합니다. (버퍼 크기 \* x 번째 주기마다 기록 \* 작업 간격) 수식을 사용하여 최소 기록 시간 프레임을 추정할 수 있습니다.

**기록 조건:(Record condition:)** 여기에 부울 변수, 값 또는 부울 식을 입력할 수 있습니다. 이 조건이 충족되는 즉시 TRUE 값을 받고, 값 기록이 시작됩니다. 아무 값도 입력하지 않으면 트레이스 구성이 컨트롤에 로드되고 응용 프로그램이 실행되는 즉시 값 기록이 시작됩니다.

## 첫 단계 - 모션

**런타임 버퍼 크기:(Runtime buffer size):** 현재 구성된 레코드에서 트레이스 변수 값을 수집하기 위해 런타임 시스템에 현재 제공된 링 버퍼의 크기(바이트)입니다. 그런 다음 트레이스 창에 값 곡선으로 버퍼 콘텐츠가 표시됩니다. 기본값: 100.



구성이 변경되면 트레이스 객체를 다시 기록해야 합니다.

디스플레이 설정(표시)...

이제 디스플레이 설정을 입력할 수 있습니다. 이러한 설정은 다시 로그인하지 않아도 유효합니다.

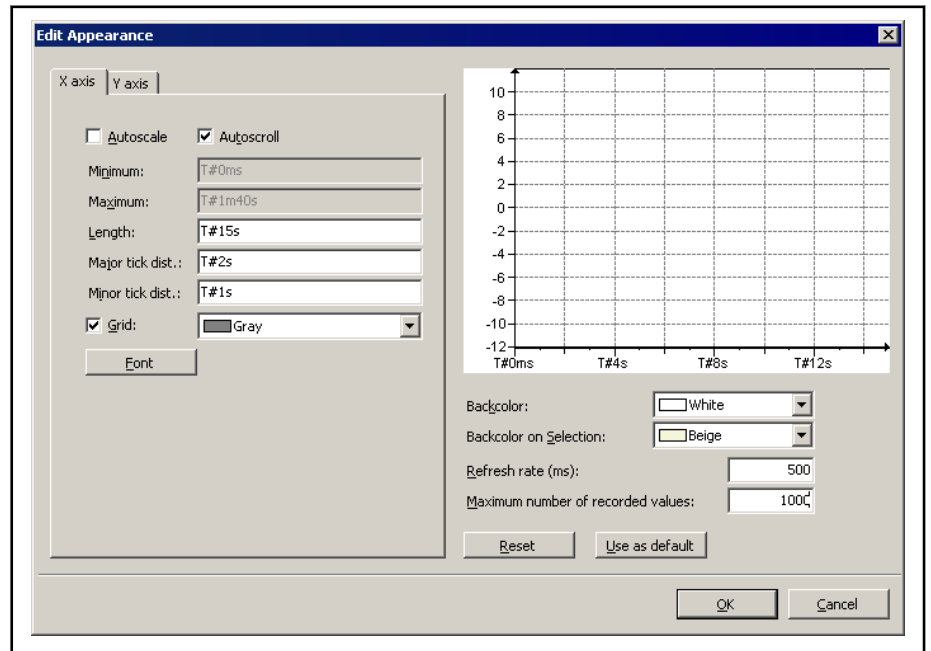


그림 5-66: 대화 상자: 디스플레이 설정(표시)

**자동 스크롤(Autoscroll)**을 선택하고 **길이(Length)**를 곡선에서 15 초 기간에 배치합니다. **스케일의 눈금폭**이 1 초인 경우 **스케일의 눈금폭**이 2 초 그리드로 설정됩니다.



정밀한 프로세스를 위해서는 15 초를 초과하면 안 됩니다.

새 변수...

이제 필수 변수의 이름을 지정하고 이러한 변수의 화면 표시를 조정해야 합니다.

## 첫 단계 - 모션

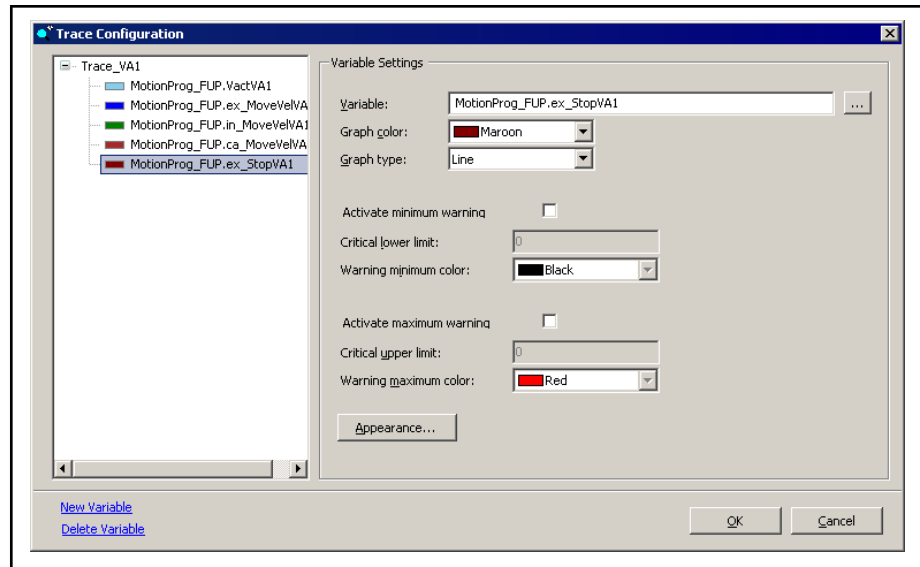


그림 5-67: 대화 상자: 새 변수 만들기 및 화면 표시 조정

색(Color) 및 그래프 유형(Graph type)은 변수(Variables) 곡선과 관련이 있습니다.

또한 각 한계가 설정된 경우 최소값 이하로 떨어지거나 최대값을 초과하는 곡선의 부분은 나머지 곡선 부분에 대해 색으로 오프셋될 수 있습니다.

트레이스 기록을 제어할 수 있는 메뉴:

트레이스 창의 왼쪽 섹션을 마우스 오른쪽 버튼으로 클릭하면 상황에 맞는 메뉴가 나타납니다.

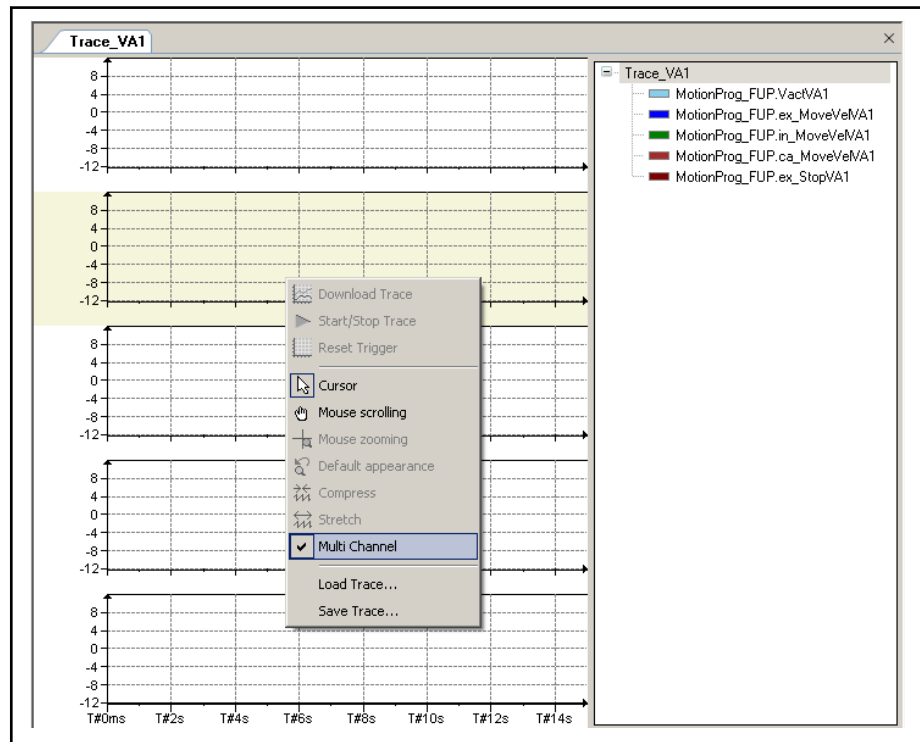


그림 5-68: 트레이스 기록을 제어할 수 있는 메뉴

**트레이스에 로그인(Login to Trace):** 실행 중인 응용 프로그램에 대해 컨트롤에 구성된 트레이스 레코드를 활성화하려면 먼저 "트레이스 응용 프로그램"을 명시적으로 로드해야 합니다.

## 첫 단계 - 모션

**트레이스 시작/중지(Start/Stop Trace):** 이 명령은 트레이스 기록 표시를 시작하거나 중지합니다.

기록이 현재 중지된 경우 이 명령 앞에는  아이콘이 표시됩니다.

기록이 현재 실행 중인 경우 이 명령 앞에는  아이콘이 표시되고, 이때 이 명령을 실행하면 기록이 중지됩니다. 즉, 트레이스 창에 현재 표시된 변수 값 곡선의 색선이 "동결"됩니다.

**다중 채널(Multi Channel):** 이 명령을 사용하면 변경할 트레이스 곡선을 볼 수 있습니다. 기본적으로 모든 변수의 곡선이 동일한 차트에 표시됩니다. 다중 채널 보기에서는 각 변수의 곡선이 동일한 X 축을 사용하여 자체 다이어그램에서 시각화됩니다. 동시에 확대/축소하고 스크롤하면 모든 다이어그램의 X 축에 반영됩니다.

**커서(Cursor):** 이 명령이 활성화되어 있으면(기본값) 트레이스 창의 좌표계에서의 현재 커서 위치가 항상 프로그래밍 시스템의 상태 표시줄에 표시됩니다.

**마우스 스크롤(Mouse scrolling):** 이 모드에서는 커서가 양방향 화살표로 간단히 표시되고, 트레이스 창에서 커서를 이동하면 현재 표시된 기록의 시간 축을 왼쪽 또는 오른쪽으로 이동할 수 있습니다.

**기본 표시(Default appearance):** 이 명령을 사용하면 예를 들어, 확대/축소를 통해 기록이 변경되었더라도 기록 표시를 다시 기본값으로 재설정할 수 있습니다.

### 트레이스 수행:

- 응용 프로그램에 로그인  
컨트롤에 연결합니다.
- 트레이스에 로그인  
응용 프로그램이 실행 중인 경우 "트레이스에 로그인"(Login to Trace) 상황에 맞는 메뉴를 통해 필요한 데이터를 컨트롤로 전송합니다.  
그러면 트레이스가 자동으로 기록되기 시작합니다.
- 값 쓰기  
다음 값은 동일하게 아래와 같이 설정됩니다([입력 변수의 "값 쓰기"\(Write Values\) 125 페이지](#)).
  - ex\_MoveVelVA1: TRUE 로 설정
  - ex\_StopVA1: TRUE 로 설정
- 기록 끝내기  
트레이스의 상황에 맞는 메뉴 항목 "트레이스 시작/중지"(Start/Stop Trace)를 통해 트레이스를 중지합니다.
- 트레이스 기록 평가  
기간 데이터를 설정합니다(아래 표 참조).

| 단축키                  | 트레이스 다이어그램에서의 동작                     |
|----------------------|--------------------------------------|
| <Alt>+<왼쪽/오른쪽> 화살표 키 | 모든 다이어그램에서 X 축을 가로로 스크롤              |
| <위쪽/아래쪽> 화살표 키       | 선택한 다이어그램만 가로로 스크롤                   |
| <+/-> 키              | 모든 다이어그램의 X 축 및 Y 축을 따라 좌표계 동시 확대/축소 |
| <Shift>+<+/-> 키      | X 축을 따라 모든 다이어그램을 가로로 스크롤            |

## 첫 단계 - 모션

| 단축키            | 트레이스 다이어그램에서의 동작          |
|----------------|---------------------------|
| <Ctrl>+<+/-> 키 | Y 축을 따라 모든 다이어그램을 세로로 스크롤 |
| <왼쪽/오른쪽> 화살표 키 | 커서를 왼쪽/오른쪽으로 스크롤          |

표 5-2: 다중 채널이 활성화된 경우 트레이스 편집기의 단축키

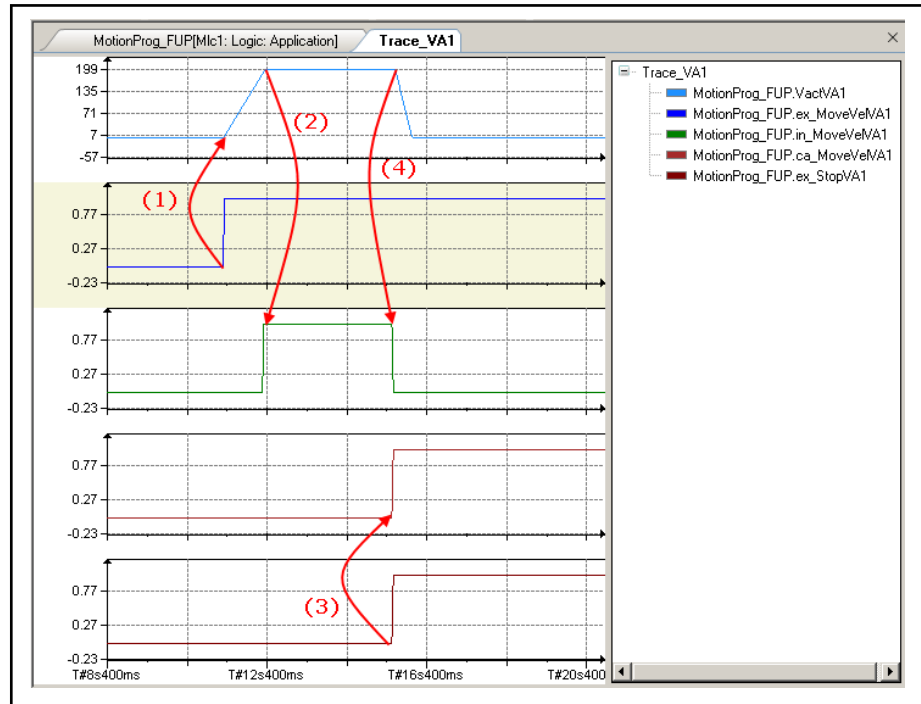


그림 5-69: 가상 축 VA1에 대한 트레이스 기록

## 트레이스 기록 평가

- ① - 가상 축 시작
- ② - 설정 속도에 도달
- ③ - 가상 축 중지
- ④ - 가상 축의 속도가 줄어듦

가상 축의 동작은 동일한 매개변수를 입력할 경우 실제 축의 동작과 유사합니다. 가속도가  $20\text{rad/s}^2$  이고 감속도가  $50\text{rad/s}^2$  일 때의 속도 특성을 비교해 보십시오.

## 5.5.5 실제 축 RA1 프로그래밍(ST(구조체언어))



지금부터는 계속 "실제" 드라이브 기술이 필요합니다.

또는

대안으로 MC\_Power 없이 두 번째 가상 축을 선택합니다.

실제 축을 사용하려면 먼저 PLCopen 평선블럭 MC\_Power의 인스턴스가 필요합니다.

그 후에는 MC\_GearIn 및 MC\_GearOut의 인스턴스가 각각 하나씩 필요합니다.

MC\_Stop의 인스턴스를 사용하여 실제 축을 중지합니다.

가상 축 VA1 프로그래밍, 프로그래밍 언어 FBD(평선블럭 다이어그램) 섹션(93 페이지)과 유사하게 프로그래밍을 실행합니다.

---

## MC\_Power

### 새 줄

MC\_Power 삽입 및 "PowerRA1" 인스턴스 선언

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Enable" → BOOL 유형의 "en\_PowerRA1"

"Axis" → "AXIS\_REF" 유형의 "MlcVarGlobal.RA1"

출력:

"Status" → BOOL 유형의 "st\_PowerRA1"

"Error" → BOOL 유형의 "er\_PowerRA1"

"ErrorID" → ERROR\_CODE 유형의 "ID\_PowerRA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "Ident\_PowerRA1"

---

## MC\_GearIn

### 새 줄

MC\_GearIn 삽입 및 "GearInRA1" 인스턴스 선언

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Execute" → BOOL 유형의 "ex\_GearInRA1"

"RatioNumerator" → 5

"RatioDenominator" → 10

"MasterFineadjust" → 20(%)

"Master" → "AXIS\_REF" 유형의 "MlcVarGlobal.VA1"

"Slave" → "AXIS\_REF" 유형의 "MlcVarGlobal.RA1"

출력:

"InGear" → BOOL 유형의 "ig\_GearInRA1"

"Active" → BOOL 유형의 "ac\_GearInRA1"

"CommandAborted" → BOOL 유형의 "ca\_GearInRA1"

"Error" → BOOL 유형의 "er\_GearInRA1"

"ErrorID" → ERROR\_CODE 유형의 "ID\_GearInRA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "Ident\_GearInRA1"

---

## MC\_GearOut

### 새 줄

MC\_GearOut 삽입 및 "GearOutRA1" 인스턴스 선언

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Execute" → BOOL 유형의 "ex\_GearOutRA1"

"Slave" → "AXIS\_REF" 유형의 "MlcVarGlobal.RA1"

출력:

"Done" → BOOL 유형의 "do\_GearOutRA1"

"Active" → BOOL 유형의 "ac\_GearOutRA1"

"Error" → BOOL 유형의 "er\_GearOutRA1"

## 첫 단계 - 모션

"ErrorID" → ERROR\_CODE 유형의 "ID\_GearOutRA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "Ident\_GearOutRA1"

**MC\_Stop****새 줄**

MC\_Stop 삽입 및 "StopRA1" 인스턴스 선언

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Execute" → BOOL 유형의 "ex\_StopRA1"

"Deceleration" → 50(rad/s<sup>2</sup>)

"Axis" → "AXIS\_REF" 유형의 "MlcVarGlobal.RA1"

**출력:**

"Done" → BOOL 유형의 "do\_StopRA1"

"Active" → BOOL 유형의 "ac\_StopRA1"

"CommandAborted" → BOOL 유형의 "ca\_StopVA1"

"Error" → BOOL 유형의 "er\_StopRA1"

"ErrorID" → ERROR\_CODE 유형의 "ID\_StopRA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "Ident\_StopRA1"

**실제 속도 "VactRA1"**

마지막으로 축 RA1 "VactRA1"(데이터 유형 REAL)의 현재 속도를 읽을 수 있는 네트워크를 구현합니다.

현재 속도는 축의 데이터 집합(공유 메모리)에서 읽어 옵니다.

MLC 를 제어할 수 있는 모든 축의 데이터는 다음 배열에 있습니다.

**GVL\_Base.AxisData: ARRAY [1..xx<sup>4</sup>] OF ML\_AXISDATA\_SM**

여기에서 축 번호는 색인으로 사용됩니다.

변수의 이름을 다음과 같이 지정합니다.

**GVL\_Base.AxisData[1].rActualVelocity\_i,**

실제 축의 경우 축 번호는 1 입니다. (VA1 을 선언하기 전에 RA1 스캔이 실행 되었으므로 축 번호 1 은 RA1 입니다.)

- 새 줄을 삽입합니다.
- VactRA1:= 을 삽입하고 선언을 완료합니다.
- 입력 보조툴을 사용하여 전역 변수(Global Variables) ▶ ML\_BASE ▶ 전역 변수리스트(Global Variable List) ▶ GVL\_Base ▶ AxisData 로 이동합니다.

4) 제어할 수 있는 축의 개수는 컨트롤 유형 및 해당 펌웨어에 따라 달라집니다(현재 L65 의 경우: 축 64 개).

첫 단계 - 모션

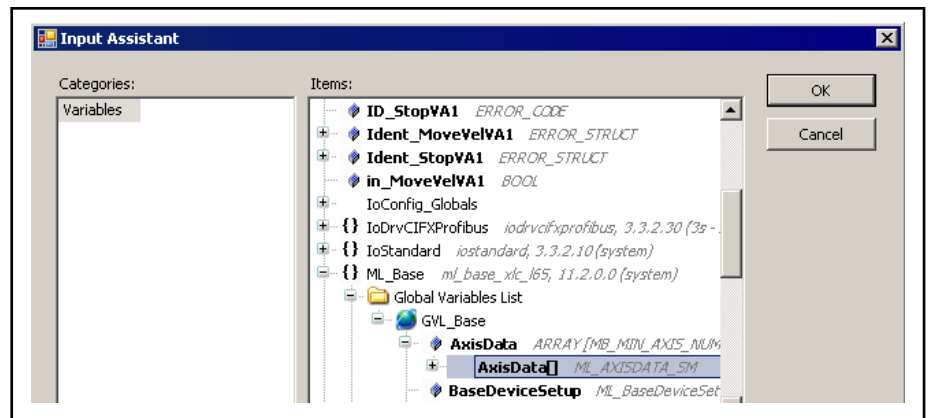


그림 5-70: ML\_Base.library 라이브러리의 전역변수리스트에 있는 실제 속도 값 VA1

Intellisense 기능을 통해 추가로 입력할 수 있습니다. 즉, "GVL\_Base.AxisData[1]."에 "."을 입력한 후 사용 가능한 모든 요소를 스크롤하여 선택할 수 있습니다.

<Enter> 키를 눌러 "rActualVelocity\_i"를 확인합니다.

- 줄 마지막에는 ";"을 입력해야 합니다.

## 첫 단계 - 모션

```

25   PowerRA1( Enable:= en_PowerRA1,
26           Axis:= MlcVarGlobal.RA1,
27           Status=> st_PowerRA1,
28           Error=> er_PowerRA1,
29           ErrorID=> ID_PowerRA1,
30           ErrorIdent=> Ident_PowerRA1);
31
32   GearInRA1( Execute:= ex_GearInRA1,
33            RatioNumerator:= 5,
34            RatioDenominator:= 10,
35            MasterFineadjust:= 20,
36            Master:= MlcVarGlobal.VA1,
37            Slave:= MlcVarGlobal.RA1,
38            InGear=> ig_GearInRA1,
39            Active=> ac_GearInRA1,
40            CommandAborted=> ca_GearInRA1,
41            Error=> er_GearInRA1,
42            ErrorID=> ID_GearInRA1,
43            ErrorIdent=> Ident_GearInRA1);
44
45   GearOutRA1( Execute:= ex_GearOut,
46            Slave:= MlcVarGlobal.RA1,
47            Done=> do_GearOut,
48            Active=> ac_GearOut,
49            Error=> er_GearOut,
50            ErrorID=> ID_GearOut,
51            ErrorIdent=> Ident_GearOut);
52
53   StopRA1( Execute:= ex_StopRA1,
54           Deceleration:= 50,
55           Axis:= MlcVarGlobal.RA1,
56           Done=> do_StopRA1,
57           Active=> ac_StopRA1,
58           CommandAborted=> ca_StopRA1,
59           Error=> er_StopRA1,
60           ErrorID=> ID_StopRA1,
61           ErrorIdent=> Ident_StopRA1);
62
63   VactRA1:= GVL_Base.AxisData[1].rActualVelocity_i;

```

그림 5-71: 실제 축 구현

## 5.5.6 전체 PLC 프로젝트 다운로드(ST(구조체언어))

프로젝트는 PLC 하위 프로젝트 가상 축 VA1 다운로드(ST(구조체언어)) 섹션 (123 페이지)에서 설명한 것과 동일하게 다운로드됩니다.

프롬프트가 표시된 후 새 프로젝트가 이전 프로젝트를 덮어씁니다.

## 5.5.7 워크플로 "가상 축 → 전자식 기어 → 실제 축"

다음 절차에서는 축과 기어를 시작하고 중지하는 방법을 보여 줍니다.

첫 단계 - 모션

1. 실제 축의 전원 켜기  
"값 쓰기"(Write Values)를 통해 "en\_PowerRA1"(MC\_Power)을 활성화합니다.  
 $st\_PowerRA1 = TRUE(MC\_Power)$
2. 가상 축 시작  
"값 쓰기"(Write Values)를 통해 "ex\_MoveVelVA1"(MC\_MoveVelocity)을 활성화합니다.  
VactVA1 = 200rpm 에서 시작이 종료됩니다.
3. 이어서 실제 축 시작  
"값 쓰기"(Write Values)를 통해 "ex\_GearInRA1"(MC\_GearIn)을 활성화합니다.  
예상되는 기어 전송 속도는 다음과 같습니다.  
 $Tr = (RatioNumerator / RatioDenominator) * (1 + MasterFineadjust)$   
VactVA1 = 200rpm 의 경우 VactRA1 = 120rpm 이 됩니다.



MasterFineadjust 값을 "%"로 지정합니다.

4. MC\_GearIn 이 "InGear"인 경우 "값 쓰기"(Write Values)를 통해 가상 축의 속도 "VcomVA1"을 변경할 수 있습니다. 변수를 두 번 클릭하고 원하는 값을 입력합니다. 실제 축이 이어서 시작됩니다.
5. "값 쓰기"(Write Values)를 통해 "ex\_GearInRA1"(MC\_GearIn)을 비활성화합니다.  
 $in\_GearInRA1 = FALSE, ac\_GearInRA1 = TRUE$   
가상 축에 이어서 실제 축이 계속해서 시작됩니다.
6. 동시 발생 종료  
"값 쓰기"(Write Values)를 통해 "ex\_GearOutRA1"(MC\_GearOut)을 활성화합니다.  
가상 축에 대한 실제 축의 종속성이 일시 중단됩니다. 그러나 두 축이 계속해서 회전하는 속도는 바뀌지 않고 그대로 유지됩니다.
7. 설정 속도 "VcomVA1" 변경  
변경된 설정 속도는 가상 축에만 영향을 미칩니다. "VactRA1"은 바뀌지 않고 그대로 유지됩니다.
8. "값 쓰기"(Write Values)를 통해 "ex\_GearOutRA1"을 비활성화합니다.  
 $do\_GearOutRA1 = FALSE$
9. 가상 축 중지  
"ex\_StopVA1"(MC\_Stop)을 활성화합니다.  
가상 축이 VactVA1 = 0rpm 으로 감속됩니다.
10. "ex\_StopVA1"(MC\_Stop)을 비활성화합니다.
11. 실제 축 중지  
"ex\_StopRA1"(MC\_Stop)을 활성화합니다.  
실제 축이 VactRA1 = 0rpm 으로 감속됩니다. 중지된 후에는 이 위치를 유지합니다.
12. "ex\_StopRA1"(MC\_Stop)을 비활성화합니다.

## 첫 단계 - 모션

실제 축의 활성 상태가 더 이상 유지되지 않습니다.

## 13. 실제 축의 전원 끄기

"값 쓰기"(Write Values)를 통해 "en\_PowerRA1"(MC\_Power)을 비활성화합니다.



RA1 에서 중지를 실행한 후 정지 모드에서만 가능합니다.

## 5.5.8 두 축의 모션 기록(트레이스), ST(구조체언어)

축 VA1 의 모션 기록(트레이스), ST(구조체언어) 섹션(127 페이지)에서 설명한 것과 동일한 방법으로 트레이스 도구를 사용하여 가상 축의 시작 및 동기화를 시각화했습니다.

다음 워크플로는 트레이스 기록을 위해 축과 기어를 시작하고 중지하는 방법을 보여 줍니다.

## 1. 실제 축의 전원 켜기

"en\_PowerRA1"(MC\_Power)을 활성화합니다.

## 2. 가상 축 시작

"ex\_MoveVelVA1"(MC\_MoveVelocity)을 활성화합니다.

## 3. 이어서 실제 축 시작

"ex\_GearInRA1"(MC\_GearIn)을 활성화합니다.

## 4. 가상 축의 속도 설정 값 증가

속도 설정 값을 증가시킵니다.

"ex\_MoveVelVA1" TRUE ⇒ FALSE ⇒ TRUE 를 통해 켵니다.

다음 그림은 트레이스 결과를 보여 줍니다.

## 첫 단계 - 모션

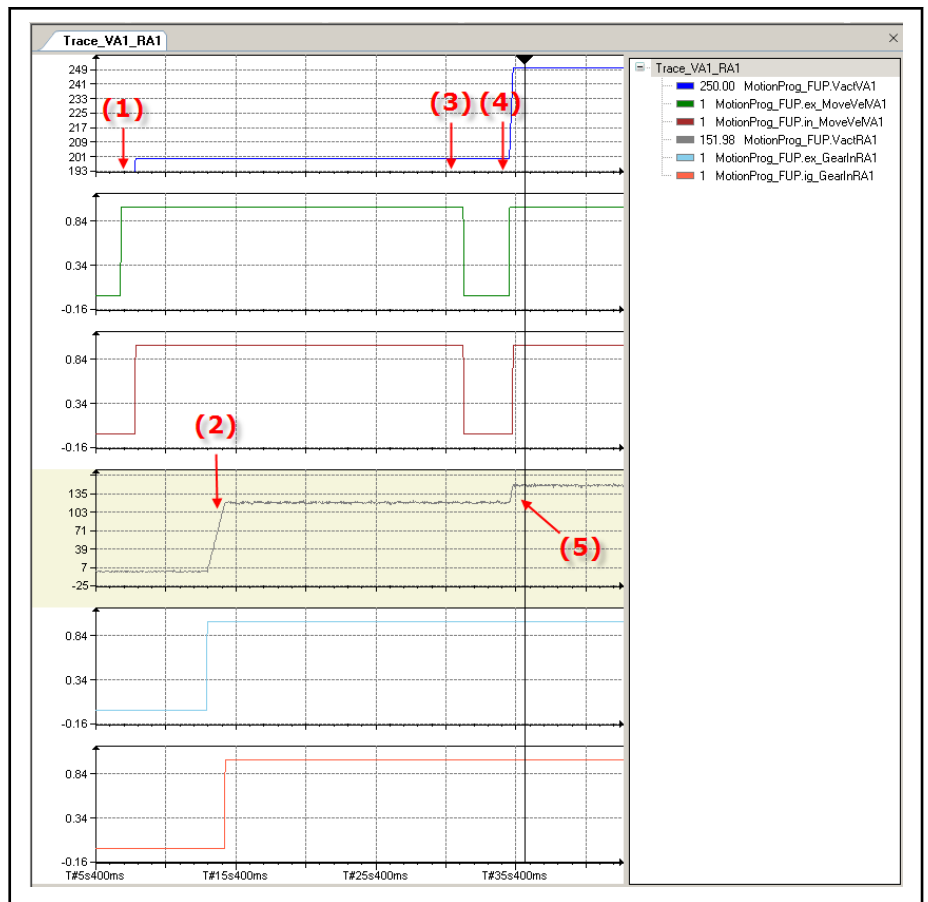


그림 5-72: 두 축의 트레이스 기록

### 트레이스 기록 평가

- ① - 가상 축 시작
- ② - 기어 기능 활성화, 실제 축 시작
- ③ - 가상 축의 설정 속도 증가, "ex\_MoveVelVA1" TRUE ⇒ FALSE
- ③ - 가상 축의 설정 속도 활성화, "ex\_MoveVelVA1" FALSE ⇒ TRUE
- ⑤ - 실제 축에 대한 반응 증가

## 5.6 프로젝트 복원

위치 만든 프로젝트를 기반으로 하여, 제어 하드웨어(예: XLC L25, L45, L65(sercos III 버스 포함) 또는 MLC L25, L45, L65) 및 드라이브 엔지니어링에 관계없이 프로젝트의 일부를 다시 생성할 수 있는 내보내기 파일이 생성되었습니다.

내보내기 파일은 "응용 프로그램"(Application) 노드부터 생성되었으며 다음과 같은 구성 요소를 포함합니다.

## 첫 단계 - 모션

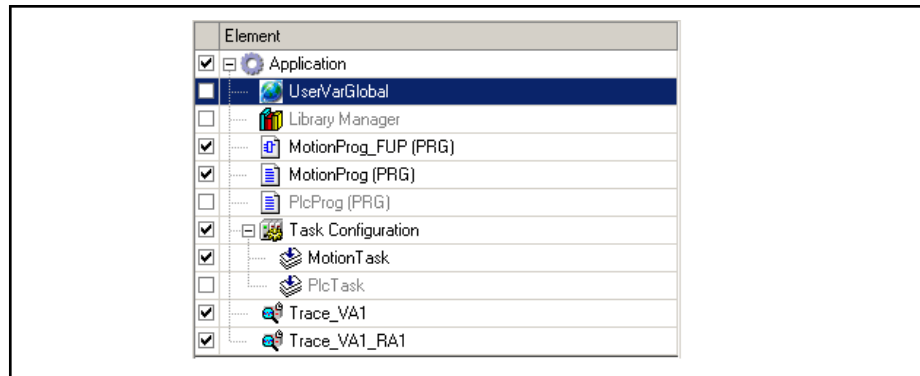


그림 5-73: 내보내기 파일의 구성 요소

가져오는 동안 제공된 구성 요소를 덮어씁니다.



내보내기 파일은 "첫 단계 - FlexProfile"의 기초로 사용할 수 있습니다.

## 복원

선택한 컨트롤(XLC L25, L45, L65(sercos III 버스 포함) 또는 MLC L25, L45, L65)에서 프로젝트가 실행되도록 하려면 약간의 사용자 개입이 필요합니다.

## 1. 컨트롤에서 이전 데이터 지우기

컨트롤의 이전 메모리 내용을 지웁니다.

자세한 내용은 [컨트롤 재설정\(CMLx5\)\(82 페이지\)](#)에 설명되어 있습니다.

컨트롤의 메모리가 지워지고 가능한 모션 구성 요소가 나타나고 PLC 구성 요소가 작동할 준비가 됩니다.

## 2. 새 프로젝트 만들기

빈 프로젝트를 새로 만듭니다. **파일(File) > 새로 만들기(New) > 프로젝트(Project)**. "First Steps 모션"(First Steps Motion)이 권장되기는 하지만 프로젝트 이름은 임의로 선택할 수 있습니다.

빈 프로젝트가 새로 생성됩니다.

## 3. 컨트롤 만들기

컨트롤(예: XLC L25, L45, L65(sercos III 버스 포함) 또는 MLC L25, L45, L65)을 제품라이브러리에서 빈 프로젝트로 끌어다 놓습니다. "DCC\_Control"이 권장되기는 하지만 컨트롤 이름은 임의로 선택할 수 있습니다.

제품유형 및 IP 주소를 지정합니다.

연결 테스트를 수행합니다.

평소와 같이 추가를 끝냅니다.

이제 프로젝트에 원하는 컨트롤이 있습니다.

## 4. 모션 하위 프로젝트 만들기

sercos III 노드에서 sercos 버스를 스캔합니다.

드라이브에 축 번호 "1"이 할당됩니다. 이름을 "RA1"로 변경해야 합니다.

## 5. 가상 축에 대해서도 같은 작업을 수행합니다. 축 번호 "2"가 할당되며, 이름을 "VA1"로 지정합니다.

전역변수리스트 "MlcVarGlobal"에는 "AXIS\_REF" 유형의 축 참조로 축 번호 및 이름이 제공됩니다.

[5.3.8 장 "모션 하위 프로젝트 다운로드" 89 페이지](#) 을 참조하십시오.

첫 단계 - 모션

6. 내보내기 파일 "Application\_FirstSteps\_Motion.iwx" 가져오기  
프로젝트 탐색기에서 컨트롤 노드를 **로직(Logic) ▶ 응용 프로그램(Application)**까지 확장합니다.  
상황에 맞는 메뉴: "응용 프로그램(Application)" **가져오기(Import)**.  
"Iw:\All Programs\Rexroth\IndraWorks\Projects"에서 내보내기 파일 "Application\_FirstSteps\_Motion.iwx"를 선택하고 "덮어쓰기" 작업을 완료합니다.  
필요한 파일이 복사되었습니다. 이제 컨트롤에서 프로젝트를 실행할 수 있습니다.



## 6 첫 단계 - FlexProfile

### 6.1 프로젝트 목표 "첫 단계 - FlexProfile"

일정 속도로 실행하도록 되어 있는 가상 축이 지정되어 있습니다. 이 축은 "FlexProfile" MotionProfile 을 통해 커플링된 실제 축의 마스터 축 역할을 합니다.

예를 들어, 실제 축에는 sercos III 케이블을 통해 컨트롤에 연결된 "IndraDrive M" 드라이브가 포함되어 있습니다.

MotionProfile 은 간단한 대기->대기(rest-in-rest) 모션을 일으키는 모션스텝으로 구성됩니다. 마스터 축이 360° 회전할 때 슬레이브 축은 180° 이동합니다. 다음 그림은 각 모션 다이어그램을 보여 줍니다.

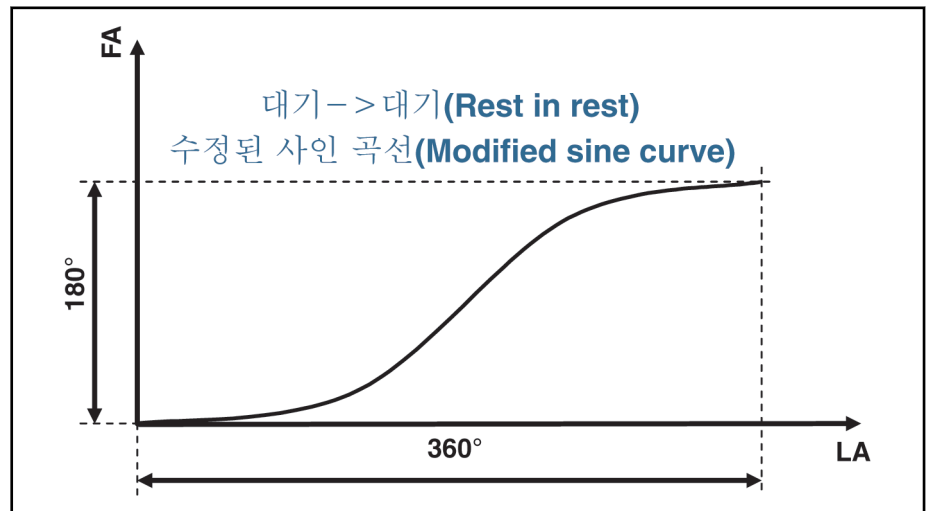


그림 6-1: 마스터 축(VA1)과 슬레이브 축(RA1) 간의 커플링 프로파일

슬레이브 축에는 각각 최대 16 개의 단계(총 64 개)로 구성된 최대 4 개의 MotionProfile 이 제공됩니다.<sup>1)</sup>

"첫 단계 - FlexProfile"에는 RA1 의 FlexProfile 0 이 사용됩니다.

FlexProfile 은 다음 두 가지 방법으로 만들 수 있습니다.

- IndraWorks, 슬레이브 축의 "FlexProfile" 대화 상자([IndraWorks 대화 상자를 통해 FlexProfile 만들기, 145 페이지 참조](#))

또는

- IndraLogic, PLC 프로그램에서 슬레이브 축으로의 데이터 전송을 위한 MB\_ChangeFlexProfileSet 평선블럭의 PLC 프로그램 코드([PLC 를 통해 FlexProfile 만들기, 148 페이지 참조](#))

ST(구조체언어)로 프로그래밍하려면 [FlexProfile, 데이터 전송\(ST\(구조체언어\)\)](#), 148 페이지를 참조하십시오.

FBD(평선블럭다이어그램)로 프로그래밍하려면 [FlexProfile, 데이터 전송\(FBD\(평선블럭다이어그램\)\)](#), 150 페이지를 참조하십시오.

실제 처리는 ML\_FlexProfile 평선블럭을 통해 시작됩니다. ML\_FlexProfile 은 슬레이브 축을 마스터 축에 연결합니다. 작업은 "첫 단계 - 모션"의 MC\_GearIn 에서 수행합니다.

ML\_FlexProfile 평선블럭은 MC\_GearOut 평선블럭이 비활성화합니다.

- ST(구조체언어)로 프로그래밍하려면 [ML\\_FlexProfile\(ST\)](#), 152 페이지를 참조하십시오.

<sup>1)</sup> FlexProfile 내의 단계 수는 펌웨어 버전 및 컨트롤의 유형에 따라 다릅니다.

## 첫 단계 - FlexProfile

- FBD(평선블럭다이어그램)로 프로그래밍하려면 [ML\\_FlexProfile\(FBD\)](#), [153 페이지](#)를 참조하십시오.

## 6.2 기술 요구 사항

- IndraMotion MLC 용 IndraWorks Engineering 이 설치된 PC, 12VRS, 컨트롤, 유형 "IndraMotion MLC L25", "IndraMotion MLC L45" 또는 "IndraMotion MLC L65", 최신 펌웨어가 로드되어 있어야 합니다.
- IP 주소를 알아야 합니다.
- 컨트롤이 이더넷을 통해 프로그래밍 장치에 연결되어 있어야 합니다. 드라이브 컨트롤러(예: 유형 "IndraDrive M"), 최신 펌웨어가 로드되어 있어야 합니다.
- 드라이브 컨트롤러가 sercos 버스를 통해 컨트롤에 연결되어 있어야 합니다.
- sercos 버스에서 드라이브 컨트롤러의 주소를 알고 있어야 하며, 드라이브가 없는 경우 브리지로 연결된 sercos 버스가 있어야 합니다.



FlexProfile 에 대한 자세한 내용은 다음 설명서를 참조하십시오.

"Rexroth IndraMotion MLC 12VRS 기능 설명" 8.7 장 "FlexProfile" (DOK-IM\*MLC-FUNC\*\*\*V12-RExx-EN-P).

FlexProfile 기능은 다음과 같은 축 유형(슬레이브 축)에서 지원합니다.

- 가상 축
- sercosDrive
- IndraDrive(컨트롤의 보간 포함)

가상 축 VA1 을 마스터 축으로 사용하고 실제 축 RA1, IndraDrive(컨트롤의 보간 포함)를 슬레이브 축으로 사용하십시오.

## 6.3 모션 하위 프로젝트 만들기

선택한 컨트롤(예: IndraLogic XLC IndraMotion L25, L45, L65(sercos III 버스 포함) 또는 MLC L25, L45, L65)에서 프로젝트가 실행되도록 하려면 약간의 사용자 개입이 필요합니다.

### 1. 컨트롤에서 이전 데이터 지우기

컨트롤의 이전 메모리 내용을 지웁니다.

자세한 내용은 [컨트롤 재설정\(CMLx5\)](#), [82 페이지](#)를 참조하십시오.

컨트롤의 메모리가 지워지고 가능한 모션 구성 요소가 나타나고 PLC 구성 요소가 작동할 준비가 됩니다.

### 2. 새 프로젝트 만들기

빈 프로젝트를 새로 만듭니다. **파일(File) ▶ 새로 만들기(New) ▶ 프로젝트(Project)**. "첫 단계 FlexProfile"(First Steps FlexProfile)이 권장되기는 하지만 프로젝트 이름은 임의로 선택할 수 있습니다.

빈 프로젝트가 새로 생성됩니다.

### 3. 컨트롤 만들기

컨트롤(예: IndraLogic XLC L25, L45, L65(sercos III 버스 포함) 또는 IndraMotion MLC L25, L45, L65)을 제품라이브러리에서 빈 프로젝트로

## 첫 단계 - FlexProfile

끝어다 놓습니다. "DCC\_Control"이 권장되기는 하지만 컨트롤 이름은 임의로 선택할 수 있습니다.

제품유형 및 IP 주소를 지정합니다.

연결 테스트를 수행합니다.

평소와 같이 추가를 끝냅니다.

이제 프로젝트에 원하는 컨트롤이 있습니다.

### 4. 모션 하위 프로젝트 만들기

sercos III 노드에서 sercos 버스를 스캔합니다.



컨트롤에서 "처리"를 선택합니다.

드라이브에 축 번호 "1"이 할당됩니다. 이름을 "RA1"로 변경해야 합니다.

### 5. 가상 축에 대해서도 같은 작업을 수행합니다. 축 번호 "2"가 할당되며, 이름을 "VA1"로 지정합니다.

전역변수리스트 "MlcVarGlobal"에는 "AXIS\_REF" 유형의 축 참조로 축 번호 및 이름이 제공됩니다.

[모션 프로젝트 다운로드\(89 페이지\)](#)가 수행될 수 있습니다. 그러면 모션 구성 요소가 "작동 준비(BB)" 상태가 됩니다.

### 6. 내보내기 파일 "Application\_FirstSteps\_Motion.iwx" 가져오기

프로젝트 탐색기에서 컨트롤 노드를 **로직(Logic) ▶ 응용 프로그램(Application)**까지 확장합니다.

상황에 맞는 메뉴: "응용 프로그램(Application)" 가져오기(Import).

"Iw:\All Programs\Rexroth\IndraWorks\Projects"에서 내보내기 파일 "Application\_FirstSteps\_Motion.iwx"를 선택하고 "덮어쓰기" 작업을 완료합니다.

필요한 파일이 복사되었습니다. 이제 컨트롤에서 프로젝트를 실행할 수 있습니다.

## 6.4 FlexProfile 만들기

### 6.4.1 FlexProfile 만들기, 일반 정보

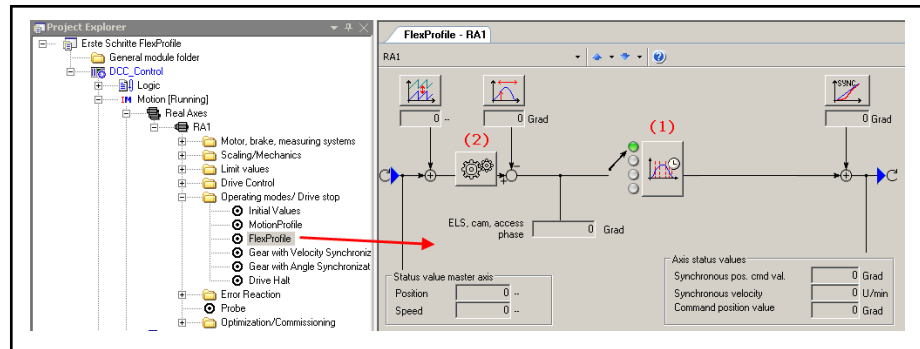


FlexProfile 을 만들거나 수정하려면 컨트롤과 PC 를 **온라인**으로 커플링해야 합니다.

### 6.4.2 IndraWorks 대화 상자를 통해 FlexProfile 만들기

프로젝트 탐색기에서 "FlexProfile" 항목인 슬레이브 축(RA1)을 클릭하며 일반 프로파일을 구성할 수 있는 대화 상자가 열립니다.

## 첫 단계 - FlexProfile



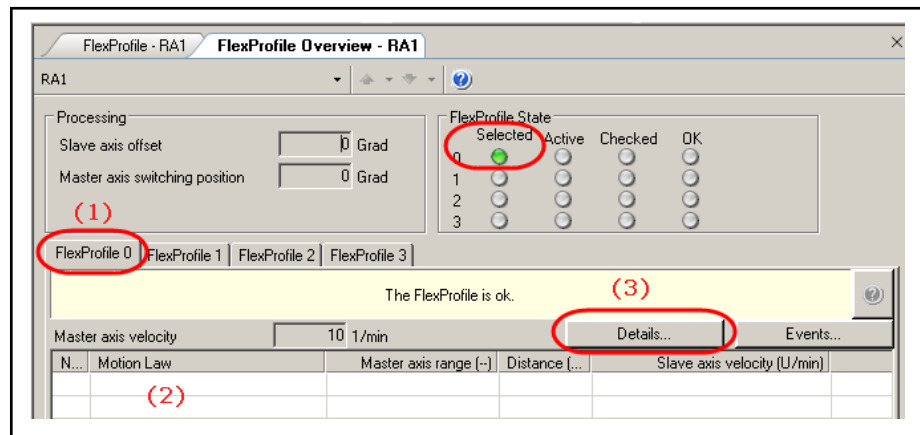
(1) "FlexProfile 0" ~ "FlexProfile 3" 대화 상자로 전환

(2) "기어 설정" 대화 상자로 전환

그림 6-2: 마스터 축과 슬레이브 축 간에 작업 체인이 있는 FlexProfile 대화 상자

(1)을 클릭하여 "FlexProfile 개요"를 엽니다.

다음 대화 상자는 축에 할당된 FlexProfile 4 개("FlexProfile 0" ~ "FlexProfile 3" 등)의 처리 상태를 보여 줍니다.



(1) "FlexProfile 0" ~ "FlexProfile 3"을 등록합니다. "FlexProfile 0"이 선택되어 있습니다.

(2) 선택한 FlexProfile(쓰기 금지)의 모션스텝 0...15(63)에 대한 개요입니다.

(3) 선택한 FlexProfile 의 세부 정보로 전환합니다.

그림 6-3: 대화 상자: "FlexProfile 개요", FlexProfile 0 이 선택됨

"세부 정보"를 통해 새 MotionProfile 을 만들 수 있습니다(1).

FlexProfile 편집기가 자동으로 열립니다.

비어 있는 구성 표가 표시됩니다. 여기에서 각 줄은 하나의 프로파일 단계 [0...15(63)]를 정의합니다.

드롭다운 메뉴에서 필요한 마스터 축 소스, 모션 법칙 및 모션스텝 유형을 선택할 수 있습니다.

아래 표시된 대로 이러한 항목을 입력하고 원하는 모션 경로에 따라 구성합니다.

첫 단계 - FlexProfile

- (1) 선택한 FlexProfile 에 대한 각 모션스텝[0...15(63)]의 구성  
(2) 전환, MotionProfile 의 데이터를 적용한 후 프로파일이 올바른지 확인  
그림 6-4: 대화 상자: FlexProfile 구성

FlexProfile 0 의 경우 위 대화 상자에 다음 데이터가 입력되었습니다.

|                                    |                                                      |
|------------------------------------|------------------------------------------------------|
| 기준(Reference)                      | 축(Axis)                                              |
| 거리(Distance)                       | 180 도                                                |
| 마스터 축 범위(Master axis range)        | 360                                                  |
| 단위(Unit)                           | -                                                    |
| 모션 법칙(Motion law)                  | 대기->대기, 수정된 사인 곡선(Rest-in-rest, modified sine curve) |
| 단계 유형(Step type)                   | 표준 단계(Standard step)                                 |
| ... 기타...                          | 0                                                    |
| 회귀점 변위(Turning point displacement) | 0.5                                                  |

FlexProfile 구성이 완료된 후에는 변경 내용이 컨트롤에 적용되어야 합니다. 변경 내용을 적용하려면 "적용 후 확인"(Apply and Check)(2)을 클릭합니다.



MotionProfile 은 확인 후 유효한 것으로 표시된 경우에만 시작할 수 있습니다.

구성이 수정되면 상태가 "확인됨"(Checked)으로 재설정됩니다.

현재 활성화된 MotionProfile 은 수정할 수 없습니다.

성공적으로 적용되면 "FlexProfile 개요" 창에서 MotionProfile 의 상태가 "확인됨"(Checked) 및 "정상"(OK)으로 표시됩니다.

## 첫 단계 - FlexProfile

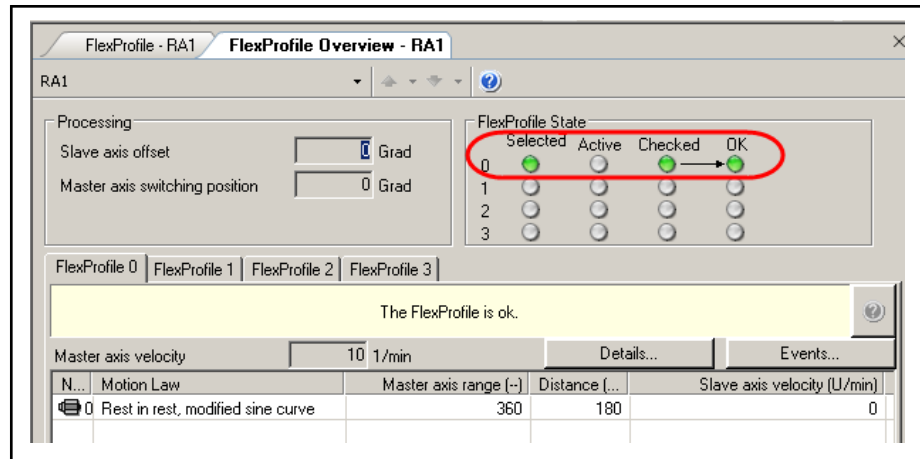


그림 6-5: 대화 상자: "FlexProfile 개요", FlexProfile 0, 확인됨 → 정상

이제 PLC 펄스블럭 "ML\_FlexProfile"과 함께 구성된 MotionProfile "FlexProfile 0"을 시작할 수 있습니다.

### 6.4.3 PLC 를 통해 FlexProfile 만들기

PLC 를 통해 FlexProfile 을 구성할 수도 있습니다.



기록하려는 프로파일의 크기 및 복잡도에 따라 "MB\_ChangeFlexProfileSet" 펄스블럭의 실행 시간이 다소 길어질 수 있습니다.

따라서 자유롭게 실행되는 PLC 프로세스에서만 펄스블럭을 사용해야 합니다.

따라서 "첫 단계 - FlexProfile"의 프로그래밍은 [PlcProg, 148 페이지](#) 또는 [PlcProg\\_FBD, 150 페이지](#) 프로그램에서 실행됩니다. 두 프로그램 모두 PlcTask 를 통해서도 제어됩니다.

실제 프로그램에는 sercos 동기 작업(예: "MotionTask")을 사용하지 않는 것이 좋습니다.

#### 해결할 작업

1. FlexProfile 4 개("FlexProfile 0"... "FlexProfile 3")의 모션스텝 0...15(63)에 필요한 데이터 준비

```
profileAxis: ARRAY[0..15] OF MB_PROFILE_STEP;
```

배열 "profileAxis"에는 FlexProfile 의 데이터가 포함되어 있습니다.

"첫 단계 - FlexProfile"의 경우 프로그램에서 "MB\_PROFILE\_STEP" 유형의 데이터 집합 "0"이 할당됩니다.

[그림 6-4 장 "대화 상자: FlexProfile 구성" 147 페이지](#) 을 참조하십시오.

2. PLC 프로그램에서 컨트롤(축 RA1)로 FlexProfile 데이터를 전송합니다.

이 작업은 "MB\_ChangeFlexProfileSet" 유형의 "ChangeFlexProfileSetRA1" 인스턴스에서 실행합니다.

[그림 6-5 장 "대화 상자: FlexProfile 개요", FlexProfile 0, 확인됨 → 정상" 148 페이지](#) 을 참조하십시오.

**FlexProfile, 데이터 전송(ST(구조체언어))**

데이터 집합을 제공할 때 먼저 "profileAxis" 변수를 선언합니다.

그런 다음 원하는 값을 사용하여 해당 구조 요소를 할당합니다.

첫 단계 - FlexProfile

구조 요소는 라이브러리 관리자 **ML\_Base\_xxx\_yyy** ▶ **데이터 유형(Data Types)** ▶ **플렉스 구조(Flex Structs)** ▶ **MB\_PROFILE\_STEP** 에 나열되어 있습니다.<sup>2)</sup>

| TYPE MB_PROFILE_STEP     |                |         |              |                                                                              |
|--------------------------|----------------|---------|--------------|------------------------------------------------------------------------------|
| Name                     | Type           | Address | Initial      | Comment                                                                      |
| Master                   | MB_MASTER_TYPE |         | MASTER_AXIS0 | Selection of master axis sources: time or position                           |
| Hub                      | LREAL          |         |              | Stroke                                                                       |
| Range                    | LREAL          |         |              | Definition of master axis section                                            |
| MotionLaw                | MC_STEP_MODE   |         |              | Definition of motion law                                                     |
| StepType                 | MB_STEP_TYPE   |         | STEP_FIX     | Segment type: standard (fixed) or Flex                                       |
| StartVel                 | LREAL          |         | 0            | Slave axis start velocities                                                  |
| StartAcc                 | LREAL          |         | 0            | Slave axis start acceleration                                                |
| StartJrk                 | LREAL          |         | 0            | Slave axis start jerk                                                        |
| EndVel                   | LREAL          |         | 0            | Slave axis final velocity                                                    |
| EndAcc                   | LREAL          |         | 0            | Slave axis final acceleration                                                |
| EndJrk                   | LREAL          |         | 0            | Slave axis final jerk                                                        |
| TravelVel                | LREAL          |         | 0            | Slave axis traveling velocity, not taken into account by all motion laws     |
| LimitAcc                 | LREAL          |         | 0            | Slave axis traveling acceleration, not taken into account by all motion laws |
| LimitJrk                 | LREAL          |         | 0            | Slave axis traveling jerk, not taken into account by all motion laws         |
| TurningPointDisplacement | LREAL          |         | 0.5          | Turning point displacement for symmetric motion laws                         |

그림 6-6: 라이브러리 관리자의 MB\_PROFILE\_STEP 구조에 대한 설명  
다음 Intellisense 기능과 함께 "." 연산자를 사용하여 입력합니다.

```

PlcProg[DCC_Control: Logic: Application]
1  PROGRAM PlcProg
2  VAR
3      profileAxis: ARRAY[0..15] OF MB_PROFILE_STEP;
4  END_VAR

1  profileAxis[0].Master:= MB_MASTER_TYPE.MASTER_AXIS0;
2  profileAxis[0].Hub:= 180;
3  profileAxis[0].Range:= 360;
4  profileAxis[0].MotionLaw:= MC_STEP_MODE.REST_IN_REST_MOD_SINE;
5  profileAxis[0].StepType:= MB_STEP_TYPE.STEP_FIX;
6  profileAxis[0].StartVel:= 0;
7  profileAxis[0].StartAcc:= 0;
8  profileAxis[0].StartJrk:= 0;
9  profileAxis[0].EndVel:= 0;
10 profileAxis[0].EndAcc:= 0;
11 profileAxis[0].EndJrk:= 0;
12 profileAxis[0].TravelVel:= 0;
13 profileAxis[0].LimitAcc:= 0;
14 profileAxis[0].LimitJrk:= 0;
15 profileAxis[0].TurningPointDisplacement:= 0.5;
    
```

그림 6-7: 값이 할당된 "profileAxis" 데이터 집합(ST(구조체언어))

17 째 줄부터는 일반적으로 "MB\_ChangeFlexProfileSet" 평선블럭의 "ChangeFlexProfileSetRA1" 인스턴스를 입력하여 데이터 배열을 컨트롤로 전송합니다.

2) ML\_Base\_xxx\_yyy: xxx 는 XLC 또는 MLC 를 나타내고, yyy 는 L25, L45 또는 L65 를 나타냅니다.

## 첫 단계 - FlexProfile

```

15  profileAxis[0].TurningPointDisplacement:= 0.5;
16
17  ChangeFlexProfileSetRA1(
18      Execute:= ex_ChangeFlexProfileSetRA1,
19      SetNumber:= SetNum,
20      NumberOfSteps:= 1,
21      MasterAxisVelocity:= 10,
22      ProfileStepAdr:= ADR(profileAxis),
23      Axis:= MlcVarGlobal.RA1,
24      Done=> do_ChangeFlexProfileSetRA1,
25      Active=> ac_ChangeFlexProfileSetRA1,
26      Error=> er_ChangeFlexProfileSetRA1,
27      ErrorID=> ID_ChangeFlexProfileSetRA1,
28      ErrorIdent=> Ident_ChangeFlexProfileSetRA1);

```

그림 6-8: 데이터 배열을 컨트롤로 전송하기 위한 평선블럭(FlexProfile 0의 경우 SetNum = 0)

데이터 배열은 입력 "ex\_ChangeFlexProfileSetRA1", TRUE/FALSE에서 "값 쓰기"(Write Values)를 통해 전송됩니다.

그러면 FlexProfile의 상태가 확인됨/정상이 됩니다.



SetNum= 2를 사용하여 "FlexProfile 2"에 쓰고 IndraWorks 대화 상자를 통해 결과를 "FlexProfile 0" 입력과 비교해 보십시오.

### FlexProfile, 데이터 전송(FBD(평선 블럭 다이어그램))

데이터 집합을 제공할 때 먼저 "profileAxis" 변수를 선언합니다.

그런 다음 원하는 값을 사용하여 해당 구조 요소를 할당합니다.

구조 요소는 라이브러리 관리자 **ML\_Base\_xxx\_yyy** ▶ 데이터 유형(Data Types) ▶ 플렉스 구조(Flex Structs) ▶ **MB\_PROFILE\_STEP**에 나열되어 있습니다.<sup>3)</sup>

| Inputs/Outputs   Documentation |                |         |              |                                                                              |
|--------------------------------|----------------|---------|--------------|------------------------------------------------------------------------------|
| TYPE MB_PROFILE_STEP           |                |         |              |                                                                              |
| Name                           | Type           | Address | Initial      | Comment                                                                      |
| Master                         | MB_MASTER_TYPE |         | MASTER_AXIS0 | Selection of master axis sources: time or position                           |
| Hub                            | LREAL          |         |              | Stroke                                                                       |
| Range                          | LREAL          |         |              | Definition of master axis section                                            |
| MotionLaw                      | MC_STEP_MODE   |         |              | Definition of motion law                                                     |
| StepType                       | MB_STEP_TYPE   |         | STEP_FIX     | Segment type: standard (fixed) or Flex                                       |
| StartVel                       | LREAL          |         | 0            | Slave axis start velocities                                                  |
| StartAcc                       | LREAL          |         | 0            | Slave axis start acceleration                                                |
| StartJrk                       | LREAL          |         | 0            | Slave axis start jerk                                                        |
| EndVel                         | LREAL          |         | 0            | Slave axis final velocity                                                    |
| EndAcc                         | LREAL          |         | 0            | Slave axis final acceleration                                                |
| EndJrk                         | LREAL          |         | 0            | Slave axis final jerk                                                        |
| TravelVel                      | LREAL          |         | 0            | Slave axis traveling velocity, not taken into account by all motion laws     |
| LimitAcc                       | LREAL          |         | 0            | Slave axis traveling acceleration, not taken into account by all motion laws |
| LimitJrk                       | LREAL          |         | 0            | Slave axis traveling jerk, not taken into account by all motion laws         |
| TurningPointDisplacement       | LREAL          |         | 0.5          | Turning point displacement for symmetric motion laws                         |

그림 6-9: 라이브러리 관리자의 MB\_PROFILE\_STEP 구조에 대한 설명  
다음 Intellisense 기능과 함께 "." 연산자를 사용하여 할당을 입력합니다.

<sup>3)</sup> ML\_Base\_xxx\_yyy: xxx는 XLC 또는 MLC를 나타내고, yyy는 L25, L45 또는 L65를 나타냅니다.

## 첫 단계 - FlexProfile

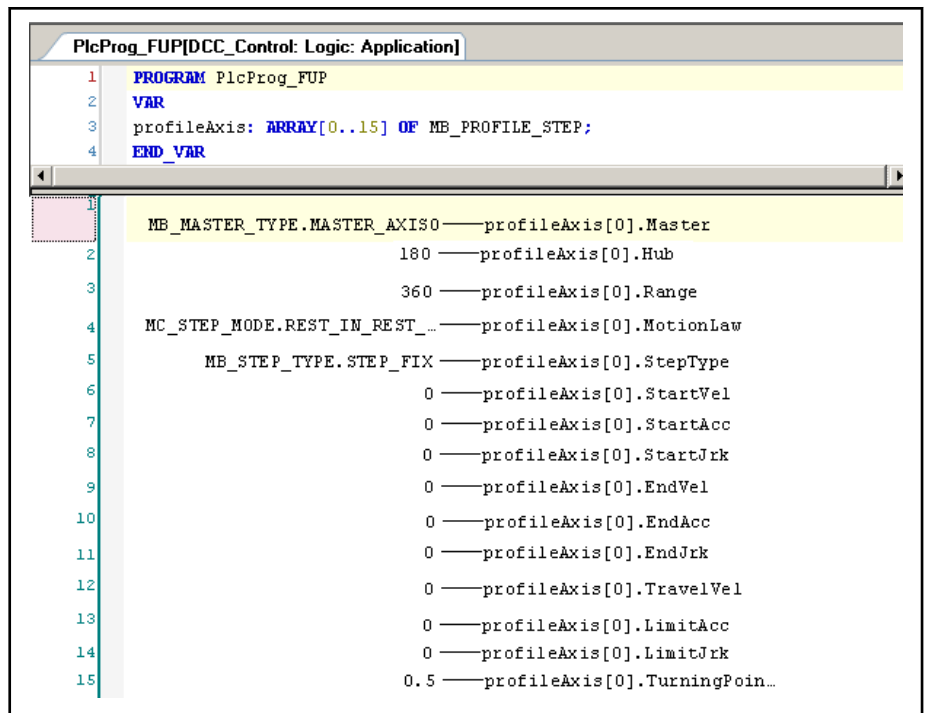


그림 6-10: 값이 할당된 "profileAxis" 데이터 집합(FBD(평선블럭 다이어그램))

일반적으로 "MB\_ChangeFlexProfileSet" 평선블럭의 "ChangeFlexProfileSetRA1" 인스턴스를 네트워크 번호 16 으로 입력하여 데이터 배열을 컨트롤로 전송합니다.

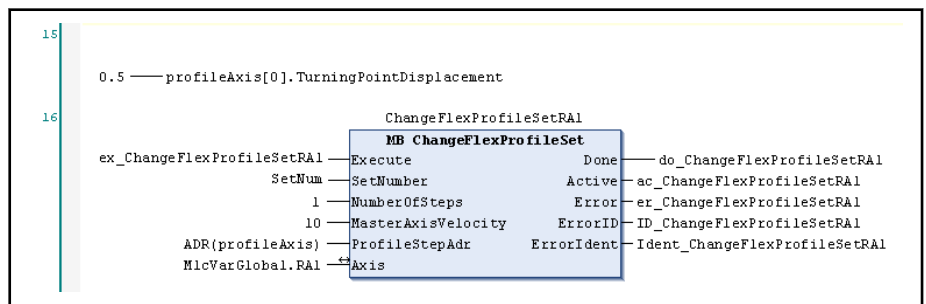


그림 6-11: 데이터 배열을 컨트롤로 전송하기 위한 평선블럭(FlexProfile 0 의 경우 SetNum = 0)

데이터 배열은 입력 "ex\_ChangeFlexProfileSetRA1", TRUE/FALSE 에서 "값 쓰기"(Write Values)를 통해 전송됩니다.

그러면 FlexProfile 의 상태가 확인됨/정상 이 됩니다.



SetNum= 1 을 사용하여 "FlexProfile 1"에 쓰고 IndraWorks 대화 상자를 통해 결과를 "FlexProfile 0" 입력과 비교해 보십시오.

## 6.5 PLC 하위 프로젝트 완료, FlexProfile 모션스텝 실행

MotionProfile 을 축에 쓴 후에 바로 "ML\_FlexProfile" 평선블럭을 실행하여 MotionProfile 을 시작할 수 있습니다. 그러면 이전에 지정하여 선택한 MotionProfile 을 통해 마스터 축과 슬레이브 축 간에 커플링이 생성되고 슬레이브 축을 동기화합니다.

이때부터 슬레이브 축 모션은 마스터 축 모션을 따릅니다. 가장 간단한 예로, PLCopen 평선블럭 "MC\_MoveVelocity"를 호출하면 마스터 축이 일정 속도로 회전합니다. 슬레이브 축이 원하는 MotionProfile 을 따라 이동합니다.

## 첫 단계 - FlexProfile

""Iw:\All Programs\Rexroth\IndraWorks\Projects"" 디렉터리에서 "Application\_FirstSteps\_IndraLogic\_Motion.iwx" 내보내기 파일을 가져오십시오(DCC\_Control ▶ 로직(Logic) ▶ 응용 프로그램(Application)의 상황에 맞는 메뉴 "가져오기"(Import)).

이 파일에는 "첫 단계 모션"(First Steps Motion)에서 이미 테스트된 프로그램(구조체언어(MotionProg) 및 FBD(평선블럭다이아그램)(MotionProg\_FBD))이 포함되어 있습니다.

다음과 같이 수정해야 합니다.

1. **VcomVA1** 을 200rpm 에서 10rpm 으로 변경합니다.

2. **MoveVelVA1:**

"가속" → 100[rad/s²]

"감속" → 100[rad/s²]

3. MC\_GearIn 의 "GearInRA1" 인스턴스를 삭제하고 ML\_FlexProfile 의 "FlexProfileRA1" 인스턴스로 바꿉니다.

**ML\_FlexProfile(ST)** 위에서 설명한 대로 "ML\_FlexProfile" 유형의 평선블럭을 추가하고 다음 설명과 같이 할당합니다.

ML\_FlexProfile 을 추가하고 "FlexProfileRA1" 인스턴스를 선언합니다.

*"MotionProg" 선언:*

1. END\_VAR 앞에 빈 줄을 삽입합니다.

2. FlexProfileRA1: 입력. <F2> 입력 도움말, 사용자 정의 유형(User-defined types) ▶ ML\_PLCOpen ▶ POU(프로그램 유닛) ▶ 모션(Motion) ▶ ML\_FlexProfile, "인수와 함께 삽입"(Insert with arguments) 옵션을 선택 취소함.

3. 다음과 같이 줄을 작성합니다.

FlexProfileRA1: ML\_FlexProfile;

---

"GearInRA1" 대신 빈 줄 추가

<F2> 입력 도움말, 인스턴스 호출(Instance calls) ▶ FlexProfileRA1, "인수와 함께 삽입"(Insert with arguments) 옵션

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Execute" → BOOL 유형의 "ex\_FlexProfileRA1"

"RatioNumerator" → 1

"RatioDenominator" → 1

"MasterFineadjust" → 0

"SetSelection" → 0

"UseSwitchingPosition" → FALSE

"SwitchingPosition" → 0

"ProfileEntry" → "MB\_PROFILE\_START\_POINT" 유형

SLAVE\_CURRENT\_MASTER\_CURRENT,

ML\_Base.library, 데이터 유형, 플렉스 구조, MB\_PROFILE\_START\_POINT

"MasterOffset" → 0

"SlaveOffset" → 0

"SyncType" → "MB\_SYNC\_TYPE" 유형

초기 값: SYNC\_RAMPIN\_CATCH\_UP, ML\_Base.library

첫 단계 - FlexProfile

"SyncVelocity" → 100

"SyncAcceleration" → 100

"ExecutionMode" → "MB\_EXECUTION\_MODE" 유형

초기 값: EXECUTE\_CYCLIC, ML\_Base.library

"Master" → "AXIS\_REF" 유형의 "VA1"

("변수", 응용 프로그램, MlcVarGlobal 영역에서 이미 선언된 변수)

"Slave" → "AXIS\_REF" 유형의 "RA1"

("변수", 응용 프로그램, MlcVarGlobal 영역에서 이미 선언된 변수)

출력:

"InSync" → BOOL 유형의 "is\_FlexProfileRA1"

"Active" → BOOL 유형의 "ac\_FlexProfileRA1"

"CommandAborted" → BOOL 유형의 "ca\_FlexProfileRA1"

"Error" → BOOL 유형의 "er\_FlexProfileRA1"

"ErrorID" → ERROR\_CODE 유형의 "ID\_FlexProfileRA1"

"ErrorIdent" → ERROR\_STRUCT 유형의 "dent\_FlexProfileRA1"

```

30     ErrorIdent=> Ident_PowerRA1);
31 FlexProfileRA1(
32     Execute:= ex_FlexProfileRA1,
33     RatioNumerator:= 1,
34     RatioDenominator:= 1,
35     MasterFineadjust:= 0,
36     SetSelection:= 0,
37     UseSwitchingPosition:= FALSE,
38     SwitchingPosition:= 0,
39     ProfileEntry:= MB_PROFILE_START_POINT.SLAVE_CURRENT_MASTER_CURRENT,
40     MasterOffset:= 0,
41     SlaveOffset:= 0,
42     SyncType:= MB_SYNC_TYPE.SYNC_RAMPIN_CATCH_UP,
43     SyncVelocity:= 100,
44     SyncAcceleration:= 100,
45     ExecutionMode:= MB_EXECUTION_MODE.EXECUTE_CYCLIC,
46     Master:= MlcVarGlobal.VA1,
47     Slave:= MlcVarGlobal.RA1,
48     InSync=> is_FlexProfileRA1,
49     Active=> ac_FlexProfileRA1,
50     CommandAborted=> ca_FlexProfileRA1,
51     Error=> er_FlexProfileRA1,
52     ErrorID=> ID_FlexProfileRA1,
53     ErrorIdent=> Ident_FlexProfileRA1);
54 GearOutRA1(
55     Execute:= ex_GearOutRA1,

```

그림 6-12: "ML\_FlexProfile"이 삽입 및 할당됨(ST(구조체언어))

ML\_FlexProfile(FBD)

위에서 설명한 대로 "ML\_FlexProfile" 유형의 평선블럭을 추가하고 다음 설명과 같이 할당합니다.

ML\_FlexProfile 을 추가하고 "FlexProfileRA1" 인스턴스를 선언합니다.

필요한 경우 입력을 할당하고 입력 변수를 선언합니다.

"Execute" → BOOL 유형의 "ex\_FlexProfileRA1"

"RatioNumerator" → 1

"RatioDenominator" → 1

## 첫 단계 - FlexProfile

"MasterFineadjust" → 0  
 "SetSelection" → 0  
 "UseSwitchingPosition" → FALSE  
 "SwitchingPosition" → 0  
 "ProfileEntry" → "MB\_PROFILE\_START\_POINT" 유형  
 SLAVE\_CURRENT\_MASTER\_CURRENT,  
 ML\_Base.library, 데이터 유형, 플렉스 구조, MB\_PROFILE\_START\_POINT  
 "MasterOffset" → 0  
 "SlaveOffset" → 0  
 "SyncType" → "MB\_SYNC\_TYPE" 유형  
 초기 값: SYNC\_RAMPIN\_CATCH\_UP, ML\_Base.library  
 "SyncVelocity" → 100  
 "SyncAcceleration" → 100  
 "ExecutionMode" → "MB\_EXECUTION\_MODE" 유형  
 초기 값: EXECUTE\_CYCLIC, ML\_Base.library  
 "Master" → "AXIS\_REF" 유형의 "VA1"  
 ("변수", 응용 프로그램, MicVarGlobal 영역에서 이미 선언된 변수)  
 "Slave" → "AXIS\_REF" 유형의 "RA1"  
 ("변수", 응용 프로그램, MicVarGlobal 영역에서 이미 선언된 변수)

출력:

"InSync" → BOOL 유형의 "is\_FlexProfileRA1"  
 "Active" → BOOL 유형의 "ac\_FlexProfileRA1"  
 "CommandAborted" → BOOL 유형의 "ca\_FlexProfileRA1"  
 "Error" → BOOL 유형의 "er\_FlexProfileRA1"  
 "ErrorID" → ERROR\_CODE 유형의 "ID\_FlexProfileRA1"  
 "ErrorIdent" → ERROR\_STRUCT 유형의 "dent\_FlexProfileRA1"

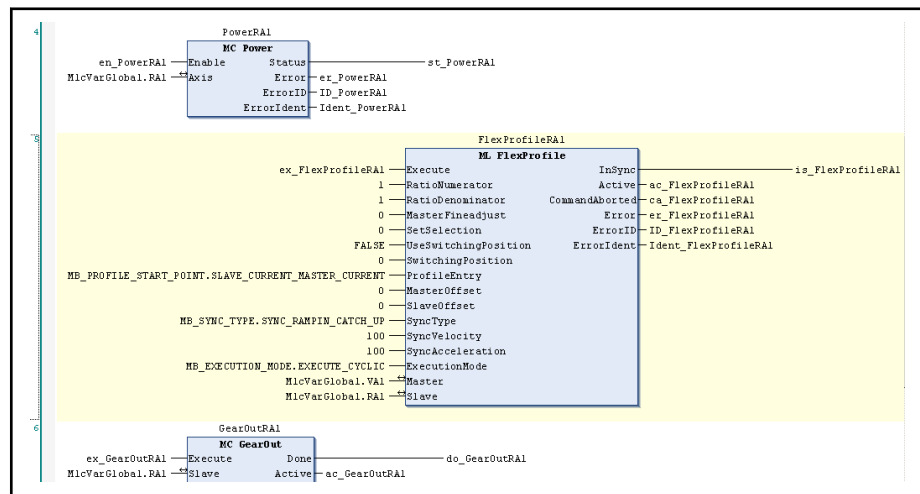


그림 6-13: "ML\_FlexProfile"이 삽입 및 할당됨(FBD(평선블록 다이어그램))

## 6.6 PLC 하위 프로젝트 다운로드 및 테스트

위에서 구현된 모션 스퀀스는 컴파일하고 컨트롤에 적용한 후 시작해야 합니다.

다음을 통해 컴파일에 액세스할 수 있습니다.

- **만들기(Create) ▶ 작성(Build):** 응용 프로그램의 추가된 부분 컴파일
- **만들기(Create) ▶ 다시 작성(Rebuild):** 전체 응용 프로그램 컴파일

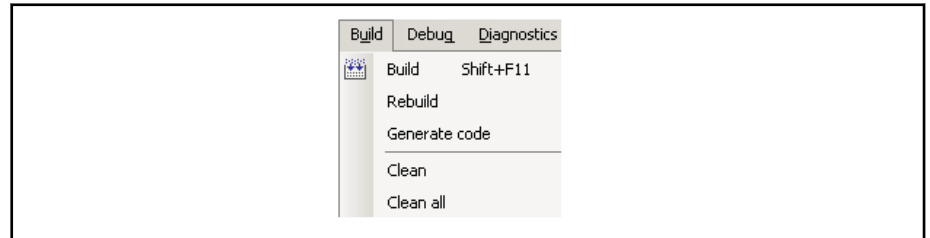


그림 6-14: 응용 프로그램 작성

컴파일된 데이터를 대상 컨트롤에 적용하려면 메뉴 항목 **디버그(Debug) ▶ 로그인(Login)**을 선택합니다. 아래 그림을 참조하십시오.

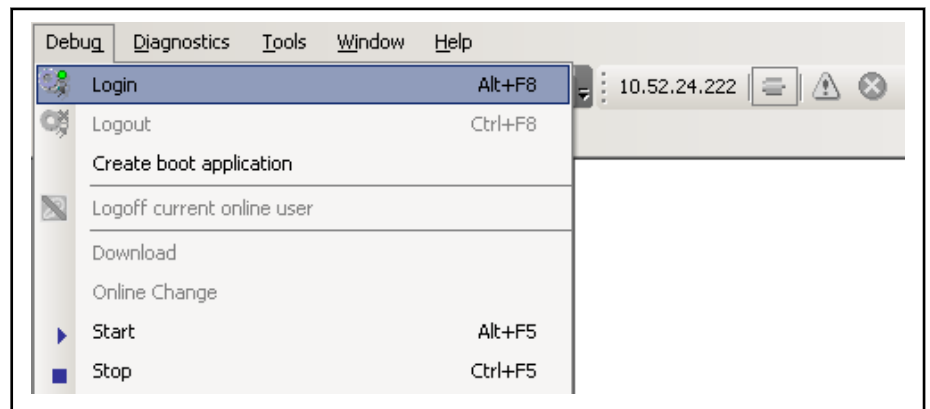


그림 6-15: 컴파일된 데이터를 컨트롤로 전송

응용 프로그램이 이미 로드되었는지 그리고 이 응용 프로그램이 현재 응용 프로그램의 이전 버전인지에 따라 다운로드 동의 여부를 묻는 메시지가 나타납니다.

응용 프로그램은 성공적인 전송 후에도 계속 중지되어 있으며(프로젝트 탐색기 및 상태 표시줄에 빨간색으로 표시됨) 값을 표시하도록 선언 및 구현에 필요한 변수가 준비됩니다. 그러나 값이 유효하지 않습니다.

**디버그(Debug) ▶ 시작(Start)**을 통해 컨트롤에서 처리를 시작합니다.

## 6.7 워크플로 "가상 축 → FlexProfile 0 → 실제 축 "

다음 절차에서는 축과 FlexProfile 을 시작하고 중지하는 방법을 보여 줍니다.

1. 실제 축의 전원 켜기

"값 쓰기"(Write Values)를 통해 "en\_PowerRA1"(MC\_Power)을 활성화합니다.

st\_PowerRA1= TRUE(MC\_Power)

2. 가상 축 시작

"값 쓰기"(Write Values)를 통해 "ex\_MoveVelVA1"(MC\_MoveVelocity)을 활성화합니다.

VactVA1 = 10rpm 에서 시작이 종료됩니다.

## 첫 단계 - FlexProfile

## 3. 이어서 실제 축 시작

"값 쓰기"를 통해 "ex\_FlexProfileRA1"(ML\_FlexProfile)을 활성화합니다.  
예상되는 커플링 동작은 다음과 같습니다.

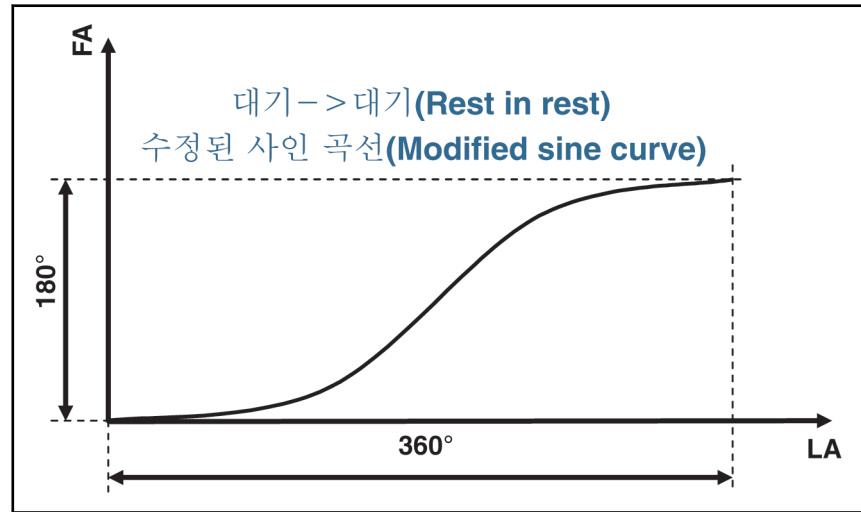


그림 6-16: 마스터 축(VA1)과 슬레이브 축(RA1) 간의 커플링 프로파일

## 4. "값 쓰기"를 통해 "ex\_FlexProfileRA1"(ML\_FlexProfile)을 비활성화합니다.

is\_FlexProfileRA1= FALSE, ac\_FlexProfileRA1= TRUE.  
가상 축에 이어서 실제 축이 계속해서 시작됩니다.

## 5. 동시 발생 종료

"값 쓰기"(Write Values)를 통해 "ex\_GearOutRA1"(MC\_GearOut)을 활성화합니다.

가상 축에 대한 실제 축의 종속성이 일시 중단됩니다. 그러나 두 축이 계속해서 회전하는 속도는 바뀌지 않고 그대로 유지됩니다.

## 6. 설정 속도 "VcomVA1" 변경

변경된 설정 속도는 가상 축에만 영향을 미칩니다. "VactRA1"은 바뀌지 않고 그대로 유지됩니다.

## 7. "값 쓰기" True/False 를 통해 "ex\_GearOutRA1"을 비활성화합니다.

do\_GearOutRA1= FALSE

실제 축이 중지합니다. 정지 상태에 도달한 후에 이 위치에서 유지됩니다.

## 8. 가상 축 중지

"ex\_StopVA1"(MC\_Stop)을 활성화합니다.

가상 축이 VactVA1 = 0rpm 으로 감속됩니다.

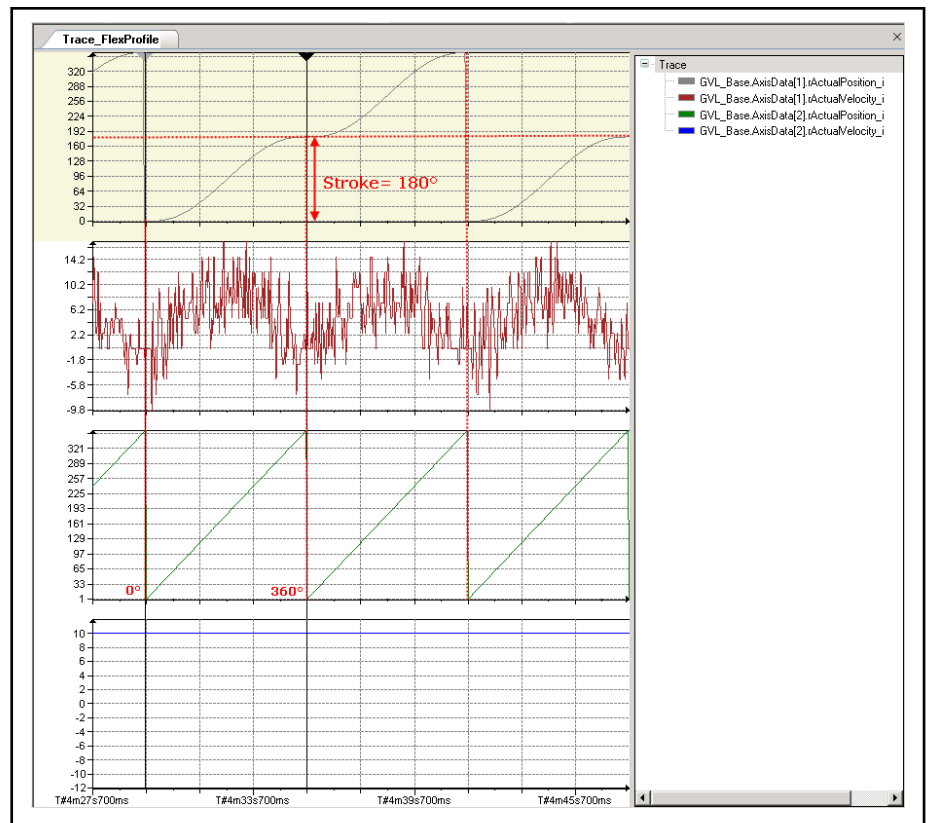
## 9. 실제 축의 전원 끄기

"값 쓰기"(Write Values)를 통해 "en\_PowerRA1"(MC\_Power)을 비활성화합니다.

## 6.8 두 축의 모션 기록(트레이스)

마스터 축 및 슬레이브 축의 모션을 트레이스 기록에 표시할 수 있습니다. 결과는 다음과 같습니다.

## 첫 단계 - FlexProfile



회색 슬레이브 축("RA1"), 위치  
 빨간색 슬레이브 축("RA1"), 속도  
 녹색 마스터 축("VA1"), 위치  
 파란색 마스터 축("VA1"), 속도  
 그림 6-17: 두 축의 모션 기록

트레이스 기록은 마스터 축의 속도 특성을 보여 줍니다.

마스터 축은 일정 속도 10[rpm]으로 회전합니다.

위는 마스터 축의 위치입니다.

맨 위의 신호는 슬레이브 축의 위치를 보여 줍니다.

이는 이전에 매개변수를 입력한 FlexProfile 이 올바르게 트레이스되었음을 나타낼 수 있습니다.

## 6.9 프로젝트 복원

만든 프로젝트를 기반으로 하여, 제어 하드웨어(IndraLogic XLC L25, L45, L65(sercos III 버스 포함) 또는 IndraMotion MLC L25, L45, L65) 및 드라이브 엔지니어링에 관계없이 프로젝트의 일부를 다시 생성할 수 있는 내보내기 파일이 생성되었습니다.

내보내기 파일은 "응용 프로그램"(Application) 노드부터 생성되었으며 다음과 같은 구성 요소를 포함합니다.

## 첫 단계 - FlexProfile

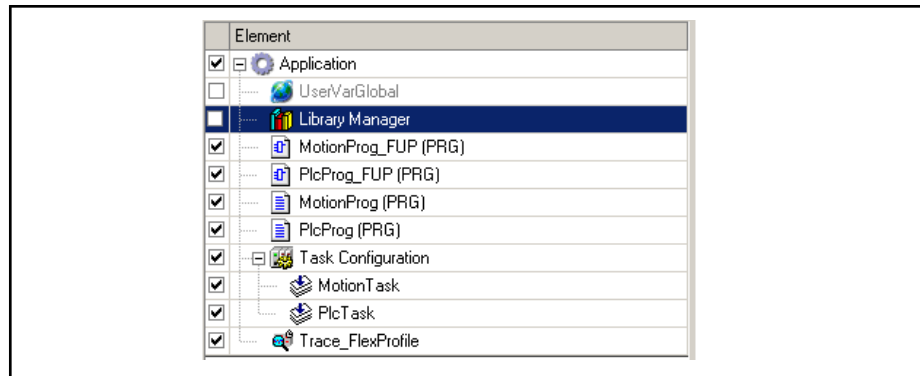


그림 6-18: 내보내기 파일의 구성 요소

가져오는 동안 제공된 구성 요소를 덮어씁니다.

"Application\_FirstSteps\_FlexProfile.iwx" 내보내기 파일은 "Iw:\All Programs \Rexroth\IndraWorks\Projects" 디렉터리에 있습니다.

## 7 첫 단계 - RobotControl

### 7.1 프로젝트 목표 "첫 단계 - RobotControl"

간단한 예제를 통해 IndraMotion MLC 를 사용하여 **RobotControl** 프로젝트(이하 RoCo 프로젝트)를 만드는 기본 절차를 보여 줍니다. IndraWorks 및 PLC 프로그래밍에 필요한 최소 구성의 키워드에 대해서도 설명합니다. 여기에서는 프로젝트를 만드는 것에서 시작하여 RoCo 소스 코드를 실행하기까지의 절차를 다룹니다.

예제 프로젝트 "RoCoTestV12"에서는 가상 축만 사용할 수 있다고 가정합니다.

실제 축을 사용할 경우의 절차에 대해서도 간단히 설명합니다.

이 경우 프로젝트의 가상 축이 실제 축으로 대체됩니다.

다음 사항을 고려하십시오.

1. 실제 축에 Axis1, Axis2 및 Axis3 과 같이 동일한 축 이름을 지정합니다.
2. 축 번호 1, 2 및 3 을 동일한 시퀀스로 할당합니다.

따라서 PLC 프로그램은 별도의 조정 없이 실행합니다.



사용자는 IndraWorks 설치 DVD 의 "AddOns\Mlc\Example-Project\RobotControl" 디렉터리에서 예제 프로젝트를 로드할 수 있습니다. 이 디렉터리에는 다음 항목이 포함되어 있습니다.

- 매개변수 내보내기 파일(\*.par)
- RCL 프로그램(\*.qll)<sup>1)</sup>
- PLC 프로그램
- 시각화 데이터

– 또는 –

다음 단계를 수행하여 차례대로 프로젝트를 만들 수 있습니다. 부록에 제공된 자세한 데이터를 사용할 수 있습니다.

### 7.2 필수 하드웨어 및 소프트웨어

필수 하드웨어 *최소 사양:*

- 컨트롤, 유형 "IndraMotion MLC Lx5"
- 최신 펌웨어가 로드되어 있어야 합니다.
- IP 주소를 알아야 합니다.
- 프로그래밍 장치(PC/랩톱)
- 컨트롤이 이더넷을 통해 프로그래밍 장치에 연결되어 있어야 합니다.

*선택 사양:*

- 실제 축 3 개(예: IndraDrive, EcoDriveCS,...)
- 최신 펌웨어(MPx07 이상)가 로드되어 있어야 합니다.
- SRAM 모듈(CFL01.1 Y1)

필수 소프트웨어 MLC 를 대상 시스템으로 사용한 IndraWorks 설치

<sup>1)</sup> RCL 은 *RobotControlLanguage* 를 의미합니다.

## 첫 단계 - RobotControl

## 7.3 RoCo 프로젝트 만들기 절차

## 1. IndraWorks 시작 및 새 프로젝트 만들기

IndraWorks 를 시작한 후에 **파일(File) ▶ 새로 만들기(File) ▶ 프로젝트(Project)**를 통해 새 프로젝트를 만듭니다.

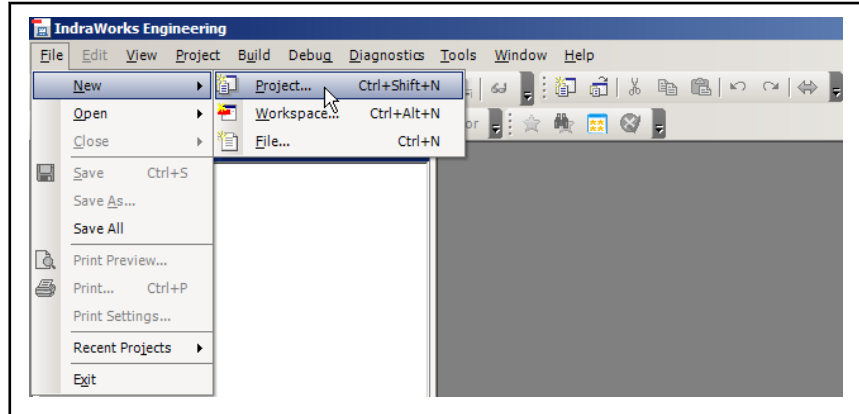


그림 7-1: 새 IndraWorks 프로젝트 만들기

## 2. MLC 컨트롤 만들기 및 구성

라이브러리의 IndraMotion MLC Lx5 를 프로젝트에 삽입하면 제품 이름, IP 주소, 하드웨어 구성 등과 같은 구성 데이터가 지정됩니다.

"구성" 대화 상자에서 평선모듈 SRAM 모듈(CFL01.1-Y1), 즉 MLC Lx5 의 메모리 확장을 만들 수 있습니다. 또한 **제품라이브러리(Device Library) ▶ FM**에서 컨트롤의 프로젝트 탐색기로 "끌어"놓기"로 가능합니다.

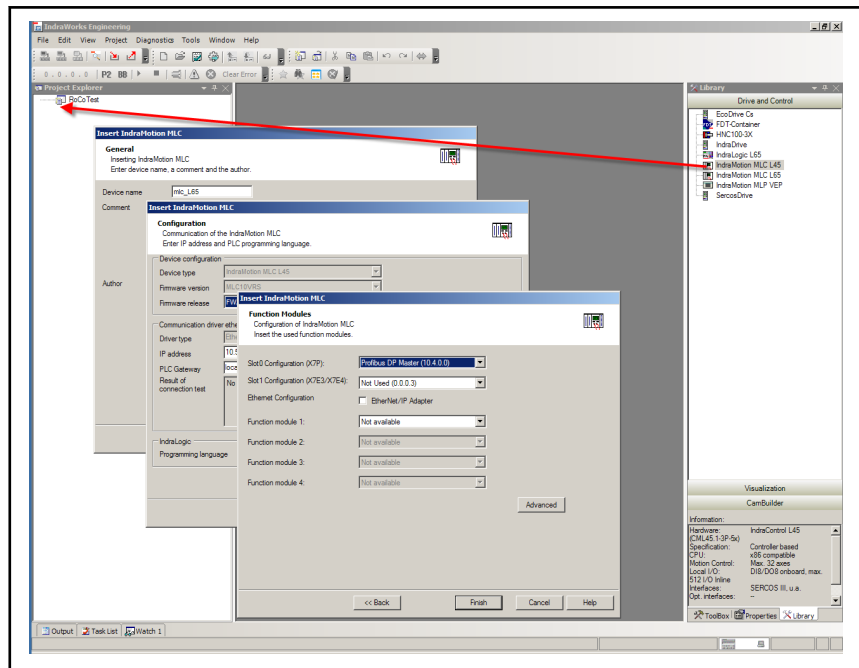


그림 7-2: MLC 만들기 및 구성

## 3. 세 개의 축 만들기 및 구성

다음 두 가지 가능한 시나리오에 대해 설명합니다.

가상 축을 사용하는 시나리오 1:

## 첫 단계 - RobotControl

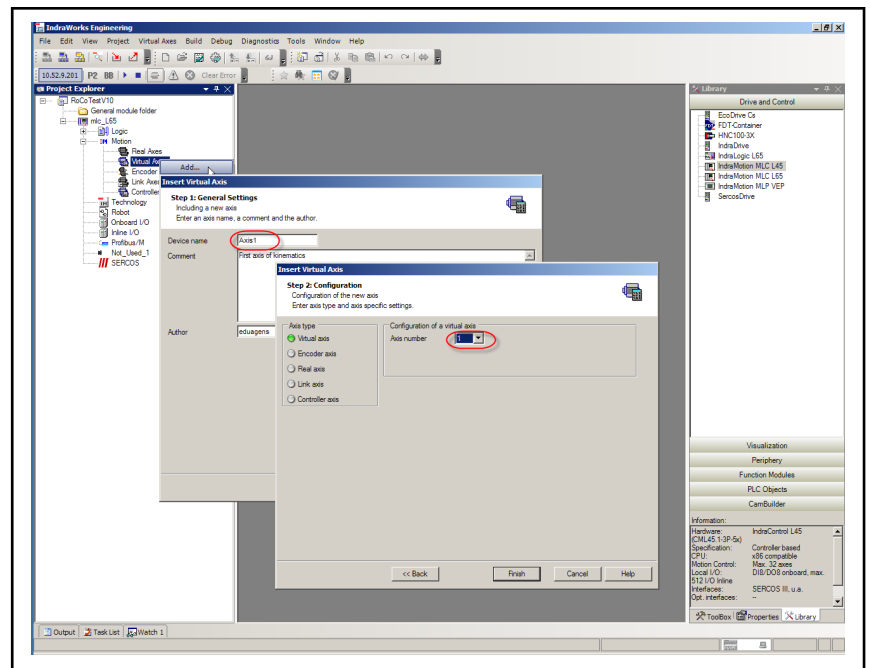


그림 7-3: 가상 축을 사용하는 프로젝트

### 실제 축을 사용하는 시나리오 2:

실제 축을 만들려면 제품라이브러리(Device Library) ▶ 드라이브 및 컨트롤(Drive and Control)에서 프로젝트 탐색기의 sercos 노드로 "끝어다" "놓습니다".

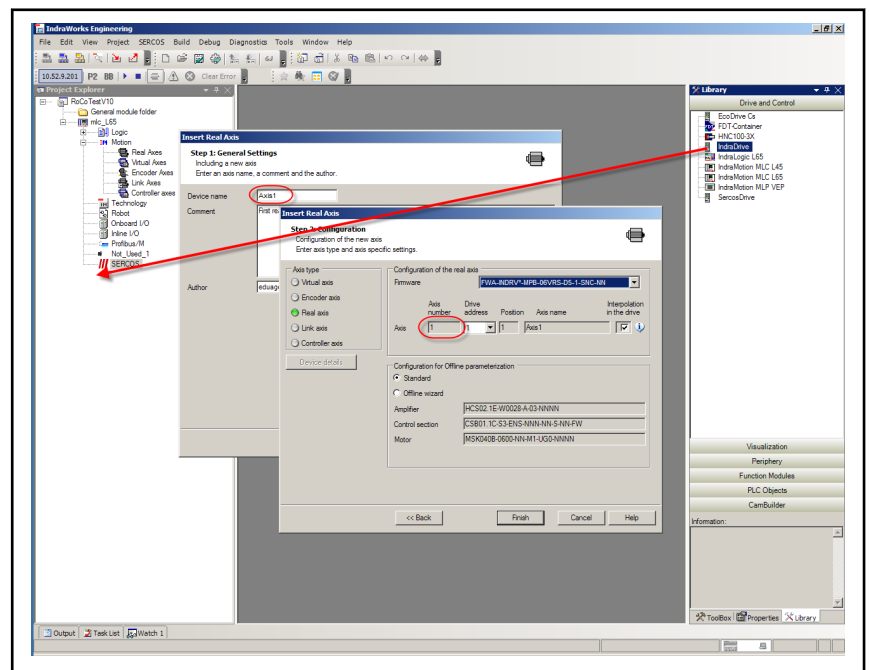


그림 7-4: 실제 축을 사용하는 프로젝트

## 첫 단계 - RobotControl



가져올 수 있는 매개변수 집합은 설명한 시나리오 둘 다에 사용할 수 있습니다. 매개변수 집합은 프로젝트 디렉터리 "RoCoTestV12\RoCoTestV12virtual.par"(가상 축의 경우) 또는 프로젝트 디렉터리 "RoCoTestV10\RoCoTestV10real.par"(실제 축의 경우)에 저장합니다.

## 4. 키네마틱 추가 및 구성

MLC 프로젝트의 키네마틱 노드에 새 키네마틱을 추가합니다.

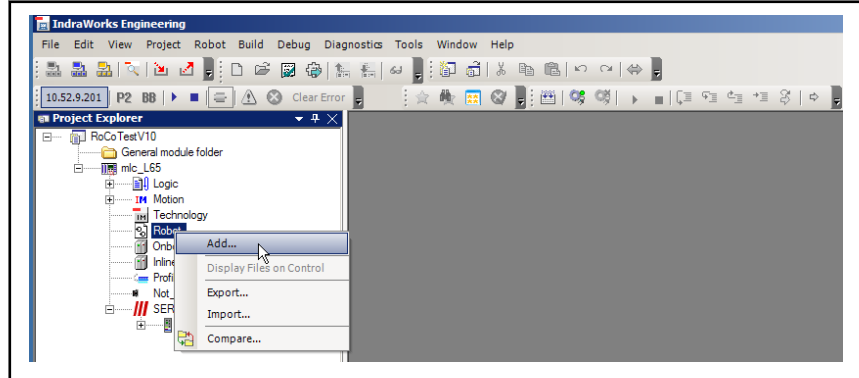


그림 7-5: 새 키네마틱 추가

다음 대화 상자에 따라 구성합니다.

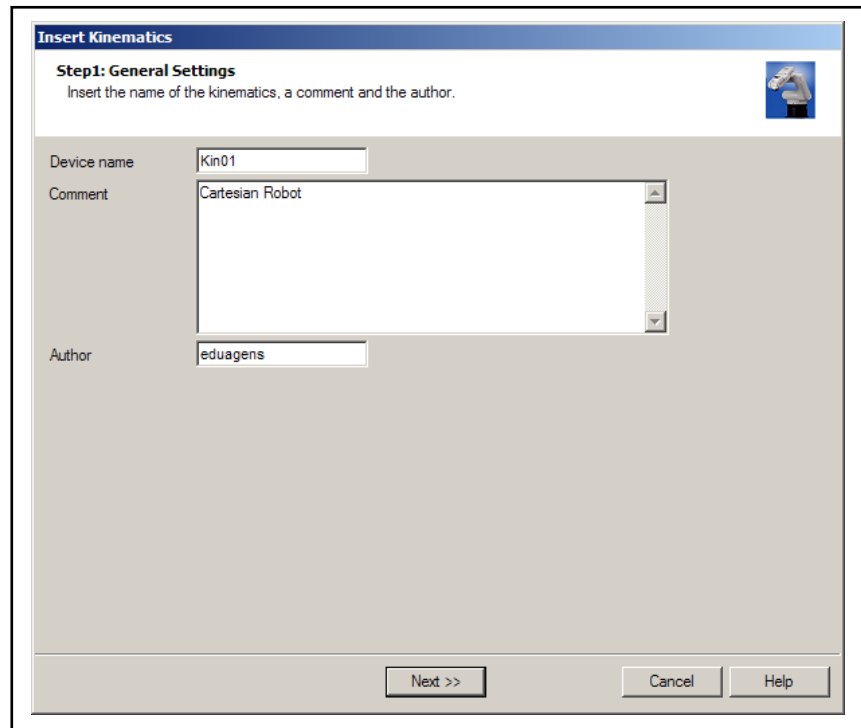


그림 7-6: 키네마틱 이름 지정



나중에 PLC 에서 AXIS\_GROUP\_REF 유형의 동일한 이름을 갖는 전역 MLC 변수를 만듭니다.

모든 키네마틱 평선블럭에는 입력 변수로 이 변수가 필요합니다.

첫 단계 - RobotControl

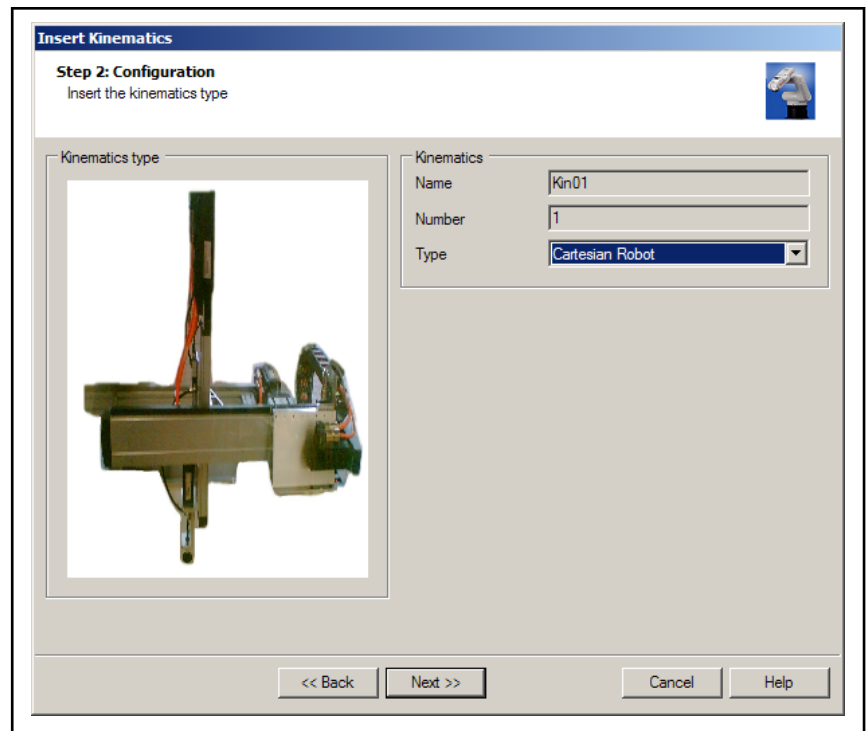


그림 7-7: 키네마틱 유형 지정

위에서 설명한 대화 상자에는 컨트롤에서 해석할 수 있는 모든 키네마틱이 나열되어 있습니다.

이 예제에서는 "직교형 로봇"(Cartesian Robot) 키네마틱 유형이 필요합니다.

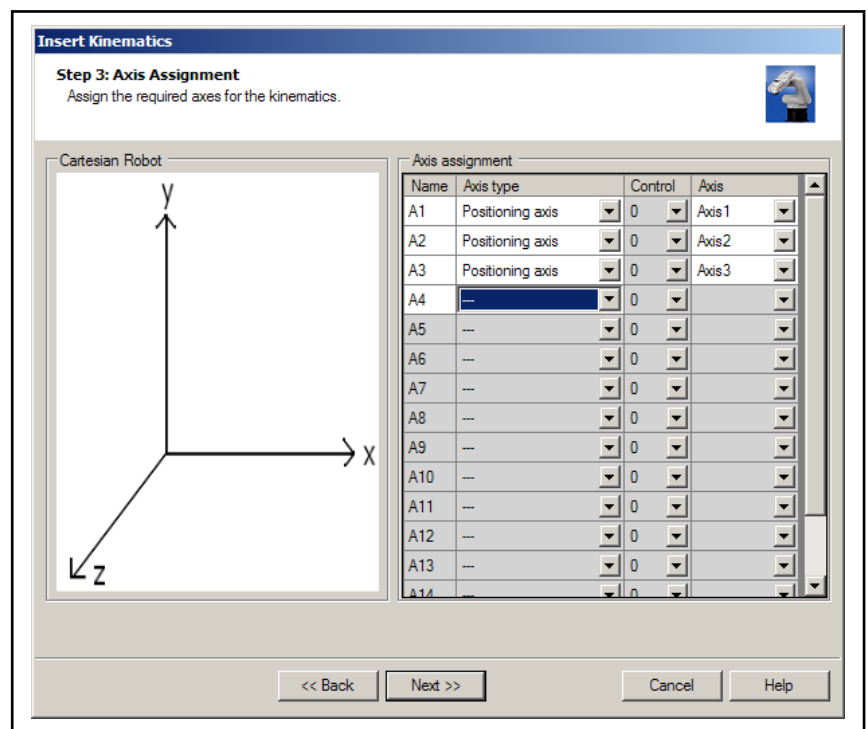


그림 7-8: 키네마틱 축 할당

## 첫 단계 - RobotControl

여기에서는 키네마틱에 속하는 모든 축(실제 또는 가상)을 정의하고 할당해야 합니다.

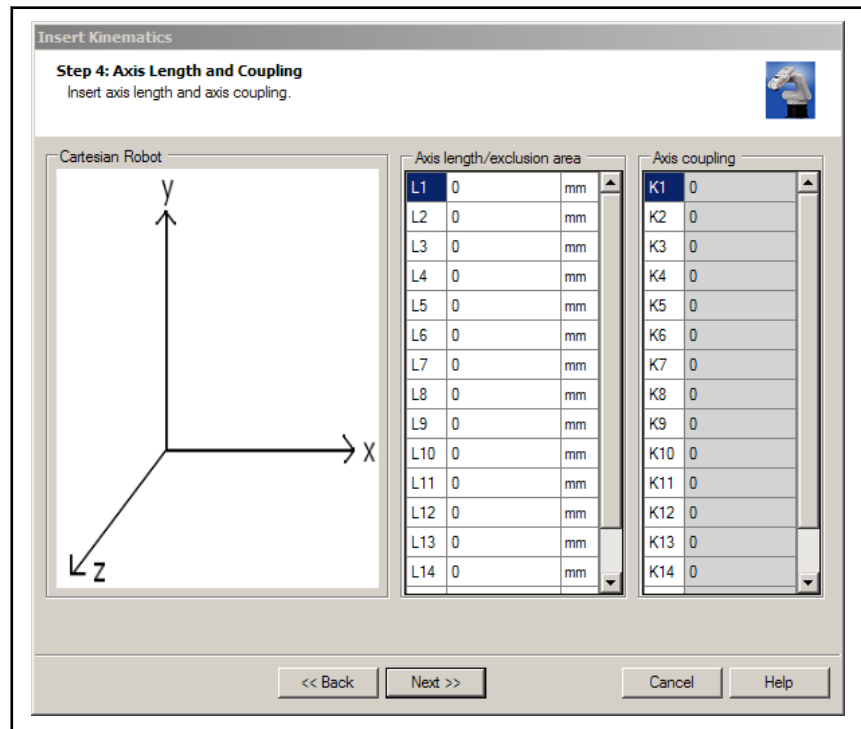


그림 7-9: 축 길이 및 축 커플링

여기에서는 로봇에 일반적인 축 커플링 및 축 길이를 입력합니다.  
설명하는 예제에서는 이 작업이 필요하지 않습니다.

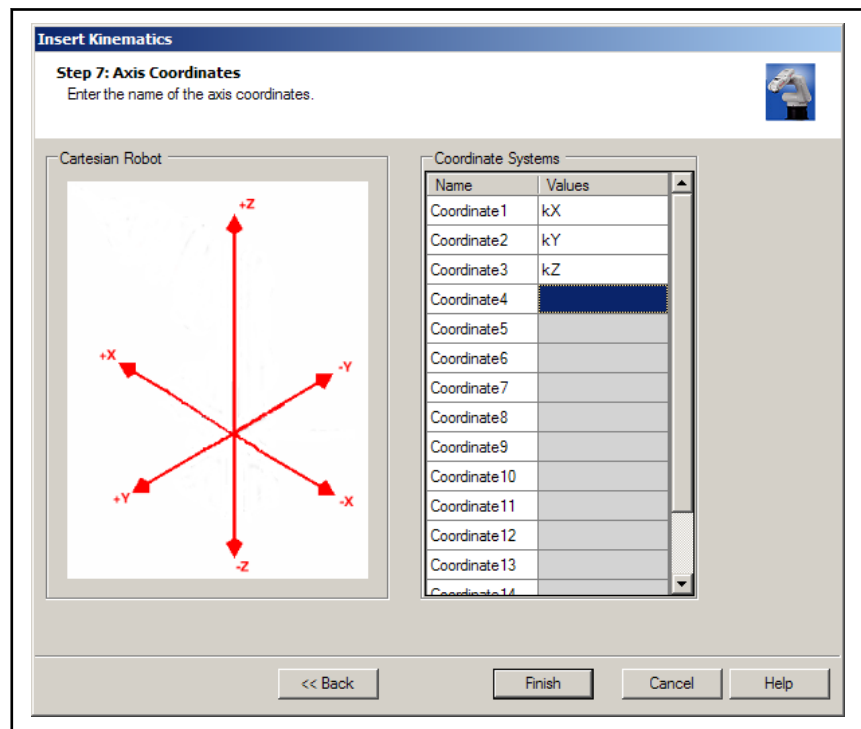


그림 7-10: 좌표 이름 지정

## 첫 단계 - RobotControl

이 단계에서는 좌표 이름을 정의합니다.



정의된 좌표 명칭은 나중에 수행되는 RoCo 프로그래밍에서 사용 됩니다.

### 5. 온라인으로 전환

온라인으로 전환하면 컨트롤이 P2 로 전환되고 지정한 설정이 컨트롤에 적용됩니다.

### 6. 매개변수 가져오기

예제 프로젝트의 경우에는 C, A 및 K 매개변수를 가져올 수 있습니다.

선택:

프로젝트 디렉터리 "RoCoTestV12\RoCoTestV12virtual.par"(가상 축의 경우) 또는 "RoCoTestV12\RoCoTestV12real.par"(실제 축의 경우)

매개변수를 가져오면 컨트롤을 BB(P4)로 전환할 수 있습니다.

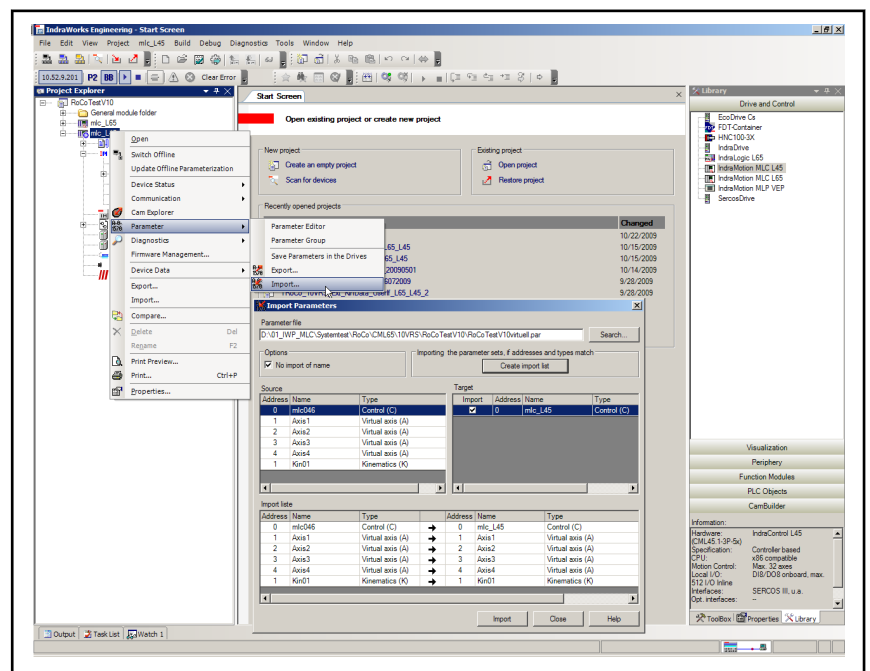


그림 7-11: 매개변수 가져오기

### 7. 모션 프로그램 만들기

키네마틱에 대해 새 RoCo 모션 프로그램을 만듭니다.

RoCo 모션 프로그램은 다음을 사용하여 RoCoTestV12 ▶ mlc\_Lx ▶ \_Emb\_Prog1.qll ▶ Prog1.qll 프로젝트에서 가져올 수도 있습니다.

- Prog1.QLL(⇒ 텍스트 전용 형식의 소스 코드)
- Prog1.PKT(⇒ 점 테이블)
- Prog1.IRD(⇒ 머신 포맷의 소스 코드, 컴파일)
- Prog1.SYM(⇒ 기호 파일, 디버깅에 필요)

## 첫 단계 - RobotControl

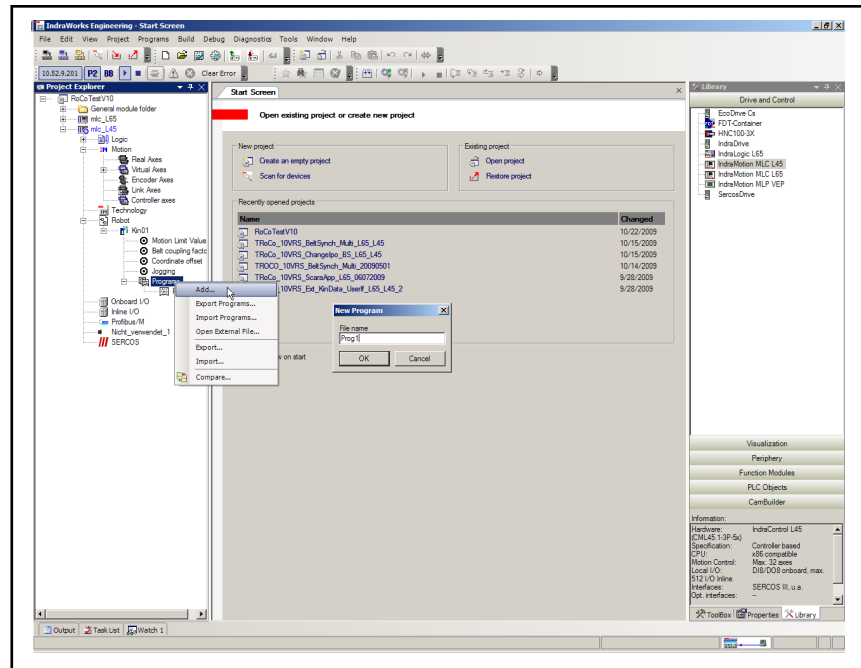


그림 7-12: 새 RoCo 프로그램 만들기

RoCo 모션 프로그램(\*.QLL)을 편집한 후에 컴파일할 수 있습니다.

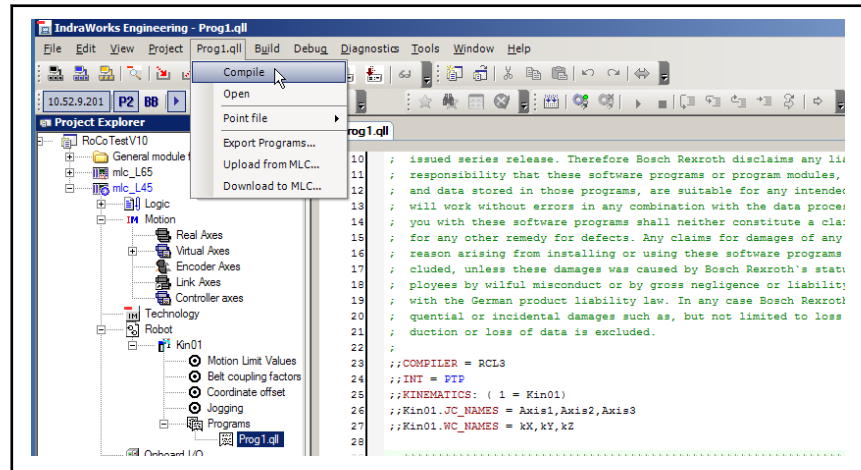


그림 7-13: RoCo 모션 프로그램 편집

컴파일 후에는 점 테이블(\*.PKT)이 자동으로 생성되고 이 테이블을 편집할 수 있습니다.

## 첫 단계 - RobotControl

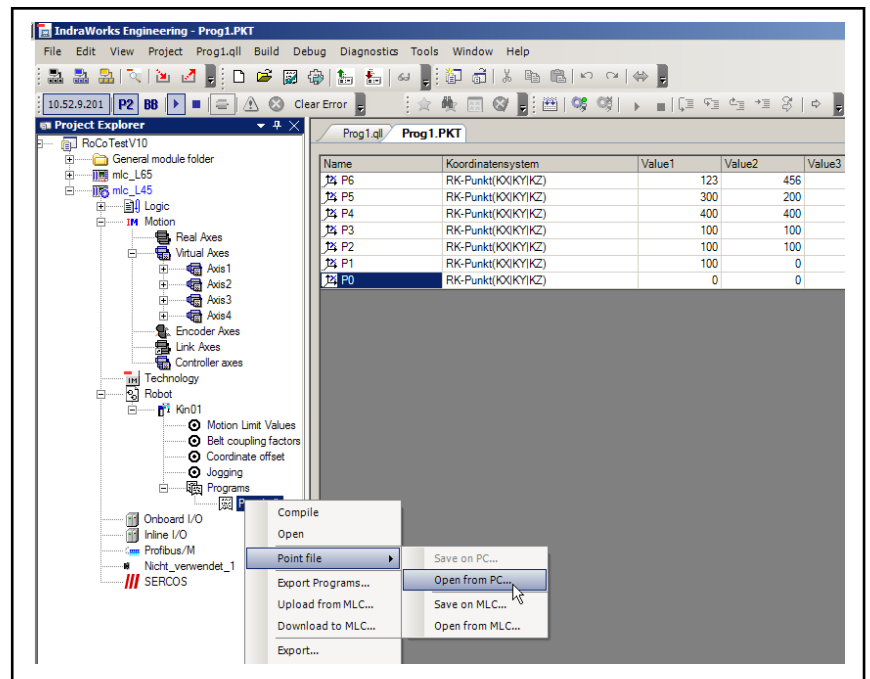


그림 7-14: 좌표 점 테이블 편집

온라인으로 작동할 경우 모션 프로그램, 점 파일 등을 컨트롤로 로드합니다.



데이터는 휘발성 메모리에 저장되므로 SRAM 평션모듈이 없는 경우에는 컨트롤이 다시 부팅된 후에 이 단계를 반복해야 합니다.

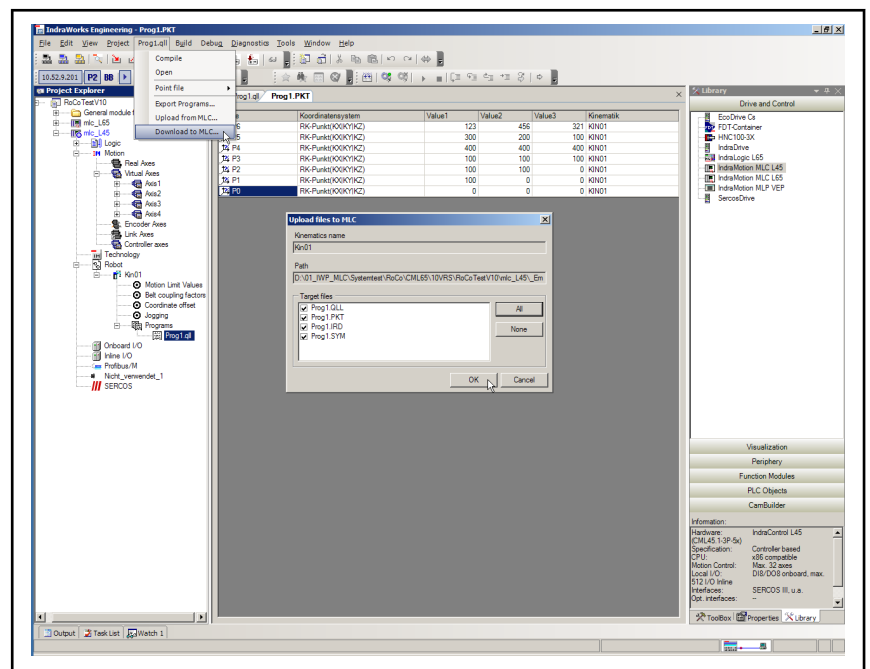


그림 7-15: 컨트롤에서 RoCo 프로그램 로드

### 8. PLC 프로그램의 RoCo 상태

예제 프로젝트에서는 PLC 부품을 제어할 수 있도록 시각화가 제공됩니다. 따라서 실제 축의 전원 공급 장치를 연결하는 것에서부터 RoCo 프로

## 첫 단계 - RobotControl

그럼 시작 및 키네마틱 상태에 이르는 모든 필요한 작동 작업도 읽을 수 있습니다.

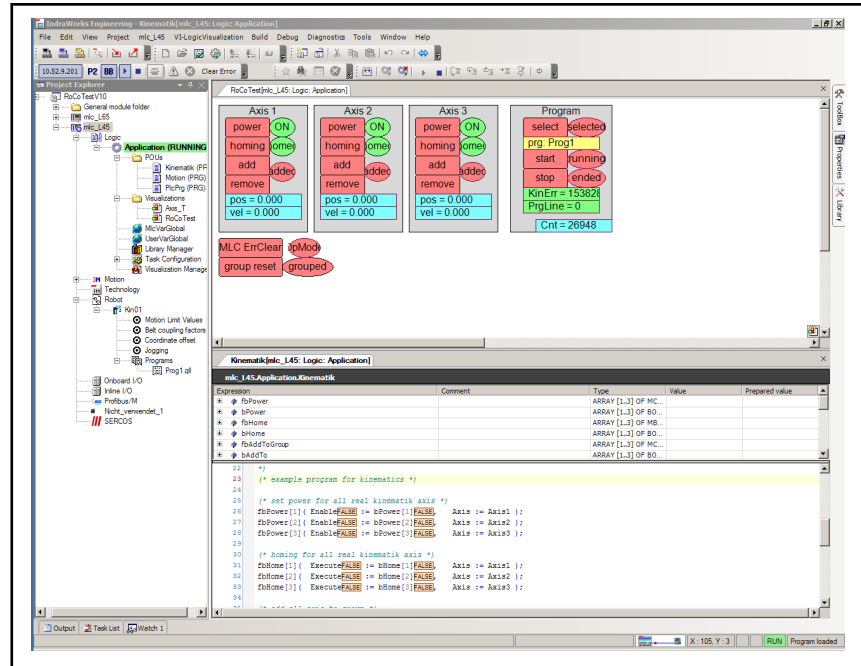


그림 7-16: PLC 프로그램의 RoCo

## 시퀀스 프로그래밍의 PLC 기능에 대한 간단한 설명

### 1. 전원 연결

이 명령은 실제 축에 대해서만 실행합니다.

이 비트는 가상 축에 대해 항상 설정되어 있습니다.

전원이 축에 연결되었는지 여부는 "AxisData[n].Axis\_Power" 비트를 사용하여 확인할 수 있습니다.

### 2. 홈잉 실행

홈잉은 절대 인코더 평가가 없는 실제 축에만 필요합니다.

축이 홈잉된 경우의 상태는 "AxisData[n].Axis\_Homed" 비트를 사용하여 확인할 수 있습니다.

### 3. 키네마틱 축 할당

이 단계에서는 정의된 키네마틱에 속한 모든 축을 그룹화합니다. 축이 축 상태 머신에서 키네마틱 상태 머신으로 변경됩니다.

그룹화된 축은 PLCopen 모션 평선블럭을 사용하여 이동할 수 없지만 RoCo 모션 프로그램에서 직접 모션 명령을 받습니다.

이는 키네마틱에 속한 모든 축이 그룹화된 경우에만 가능합니다.

조화 모션에서 모든 축이 필요한지 여부는 "AxisData[x].CoordinatedMotion" 상태에서 확인할 수 있습니다.

### 4. 키네마틱에서 축 제거

이 작업은 각 축을 RoCo 모션 프로그램 밖으로 이동하려는 경우 필요합니다.

### 5. RoCo 모션 프로그램 선택

RoCo 모션 프로그램을 실행하려면 먼저 선택해야 합니다. 이 작업은 각 프로그램이 종료 및 중지된 후에 다시 수행해야 합니다.

6. RoCo 모션 프로그램 시작  
이 명령은 선택한 프로그램을 시작합니다.
7. RoCo 모션 프로그램 중지  
모든 축 모션을 중지하고 선택한 프로그램을 재설정합니다.

#### 오류가 발생할 경우

1. 키네마틱 그룹 재설정(MC\_GroupReset)  
모든 키네마틱 오류가 삭제됩니다.
2. MLC 컨트롤 오류 및 드라이브 오류 삭제(C-0-1030)  
PLC 에서 모든 드라이브 오류를 삭제합니다.

## 7.4 "RoCoTestV12" 프로젝트에 대한 세부 정보

#### 매개변수 사전 설정(최소 변경)

C 매개변수:

C 매개변수는 프로젝트 탐색기에서 MLC 를 구현하고 가상 축을 만들 때 자동으로 할당됩니다.

IndraWorks 대화 상자 "스케일링/단위"(Scaling/Units), "설정 상태 메시지"(Settings Status Messages) 및 "모션 한계 값"(Motion Limit Values)에서 축에 유용한 변경 사항은 다음과 같습니다.

- A-0-0053 , 가속 데이터 스케일링 유형: 병진 운동
- A-0-0056 , 속도 데이터 스케일링 유형: 병진 운동
- A-0-0056, 위치 데이터 스케일링 유형: 병진 운동, 절대
- 각각 한계 값과 기간 값

C-0-2484, 키네마틱 구성 목록 및 키네마틱 구성의 모든 K 매개변수 결과

*RoCo 프로그램 목록*

```
;;COMPILER = RCL3
;;INT = PTP
;;KINEMATICS: ( 1 = Kin01)
;;Kin01.JC_NAMES = Axis1,Axis2,Axis3
;;Kin01.WC_NAMES = kX,kY,kZ

; *****
; This programme was created by:
; Author: rudokohl
; Department: esc3
; Date: 06.12.2006
;
; Programme description:...
; *****

PROGRAM Prog1
; declarations
Def Point: P0, P1, P2, P3, P4, P5, P6

BEGIN
; program
Kin01.V_PTP = 0.4 ; velocity for point to point movement
Kin01.A = 1000 ; acceleration
Kin01.V = 1200 ; velocity

;LOOP:
;JUMP LOOP

MOVE TO P0 ; move to P0
Kin01.V = 400 ; velocity
WAIT 3 ; wait for 3 seconds
REPEAT 4 TIMES
MOVE TO P1 ; move to P1
MOVE TO P2 ; move to P2
MOVE TO P3 ; move to P3
MOVE VIA P4, P5 TO P6 ; move via to P6
```

## 첫 단계 - RobotControl

```

      WAIT 3                      ; wait for 3 seconds
REPEAT_END

PROGRAM_END

```

*PLC 프로그램 목록 (키네마틱)*

PROGRAM Kinematik

VAR

```

fbPower      : ARRAY[1..3] OF MC_Power;
bPower       : ARRAY[1..3] OF BOOL;
fbHome       : ARRAY[1..3] OF MB_Home;
bHome        : ARRAY[1..3] OF BOOL;
fbAddToGroup : ARRAY[1..3] OF MC_AddAxisToGroup;
bAddTo       : ARRAY[1..3] OF BOOL;
fbRemFromGroup : ARRAY[1..3] OF MC_RemAxisFromGroup;
bRemove      : ARRAY[1..3] OF BOOL;
fbGroupReset : MC_GroupReset;
bGroupReset  : BOOL;
fbPrgSelect  : MB_IRDProgSelect;
bPrgSelect   : BOOL;
fbPrgStart   : MB_IRDProgStart;
bPrgStart    : BOOL;
fbPrgStop    : MB_IRDProgStop;
bPrgStop     : BOOL;
strProg      : STRING(8) := 'Prog1';

```

END\_VAR

(\* 키네마틱에 대한 예제 프로그램 \*)

(\* 모든 실제 키네마틱 축에 대한 힘 설정 \*)

```

fbPower[1]( Enable := bPower[1], Axis := Axis1 );
fbPower[2]( Enable := bPower[2], Axis := Axis2 );
fbPower[3]( Enable := bPower[3], Axis := Axis3 );

```

(\* 모든 실제 키네마틱 축에 대한 홈잉 \*)

```

fbHome[1]( Execute := bHome[1], Axis := Axis1 );
fbHome[2]( Execute := bHome[2], Axis := Axis2 );
fbHome[3]( Execute := bHome[3], Axis := Axis3 );

```

(\* 그룹에 모든 축 추가 \*)

```

fbAddToGroup[1]( Execute := bAddTo[1], Axis := Axis1, AxisGroup := Kin01 );
fbAddToGroup[2]( Execute := bAddTo[2], Axis := Axis2, AxisGroup := Kin01 );
fbAddToGroup[3]( Execute := bAddTo[3], Axis := Axis3, AxisGroup := Kin01 );

```

(\* 그룹에서 모든 축 제거 \*)

```

fbRemFromGroup[1]( Execute := bRemove[1], Axis := Axis1, AxisGroup := Kin01 );
fbRemFromGroup[2]( Execute := bRemove[2], Axis := Axis2, AxisGroup := Kin01 );
fbRemFromGroup[3]( Execute := bRemove[3], Axis := Axis3, AxisGroup := Kin01 );

```

(\* 프로그램 선택 \*)

fbPrgSelect( Execute := bPrgSelect, IRDProgName := strProg, AxisGroup := Kin01 );

(\* 프로그램 시작 \*)

fbPrgStart( Execute := bPrgStart, AxisGroup := Kin01 );

(\* 프로그램 중지 \*)

fbPrgStop( Execute := bPrgStop, AxisGroup := Kin01 );

(\* 그룹 재설정 \*)

fbGroupReset( Execute := bGroupReset, AxisGroup := Kin01 );

END\_PROGRAM

*PLC 프로그램 목록 (PlcPrg) 목록*

PROGRAM PlcPrg

VAR

```

fbReset      : MB_WriteParameter;
bErrClear    : BOOL;

```

END\_VAR

(\* 오류 지우기 MLC \*)

```

fbReset( Execute := bErrClear,
ParameterNumber := FP_C_0_1030,
Value := 3,
Axis := Axis1,
Done=> , Active=> , Error=> , ErrorID=> , ErrorIdent=> );

```

END\_PROGRAM

## 7.5 RobotControl - 애플리케이션 예

### 7.5.1 픽 앤 플레이스(Pick & Place) 애플리케이션의 RoCo 기본 사항

#### 일반

일반적으로 간단한 픽 앤 플레이스(Pick & Place) 애플리케이션의 공구 중심점 (이하 TCP)에 대한 모션 스퀀스는 다음과 같습니다.

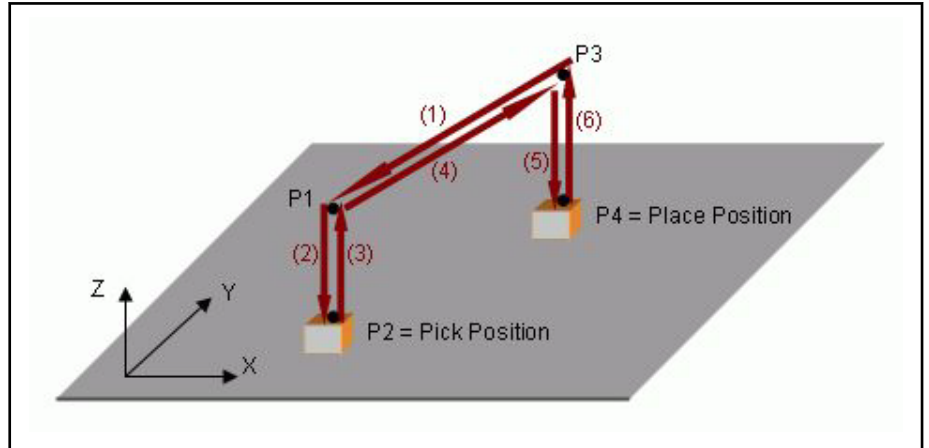


그림 7-17: 간단한 픽 앤 플레이스(Pick & Place) 애플리케이션

#### 설명

- 개체 위의 수평 모션 (1)
- 개체를 잡기 위해 개체 쪽으로 내려오는 수직모션 (2)
- 개체를 잡고 위쪽으로 올라가는 수직모션 (3)
- 개체를 놓을 위치 위의 수평 모션 (4)
- 개체를 놓을 위치 쪽으로 내려가는 수직모션 (5)
- 개체를 내려 놓은 후 개체 없이 올라가는 수직모션 (6)
- 새 개체를 대기한 후 (1)로 진행

#### 체결

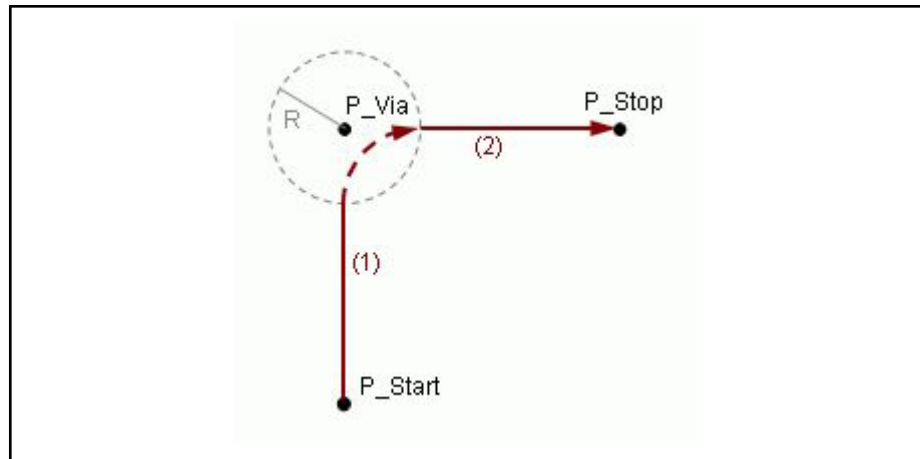
위에서 트레이스한 모션 스퀀스를 더 조화롭고 더 빠르게 수행하려면 개체 픽업과 배치 사이 또는 그 반대로 각 모션블럭을 체결할 수 있습니다.



BLOCK\_SLOPE 와 체결할 경우 프로그래밍된 체결반경 R 로 진입할 때 이후의 모션블럭이 일찍 시작됩니다. 체결반경 내에서는 두 모션 법칙이 모두 활성화됩니다. 즉, 각 모션을 겹쳐서 체결 볼 내의 결과 경로 곡선이 생성되었습니다.

체결 모션블럭의 점이 직선 위에 없으면 바로 통과하여 이동하지 않습니다.

## 첫 단계 - RobotControl



R 체결반경

그림 7-18: 일반적인 체결

수평 모션 (2)는 체결반경에 진입할 때 시작됩니다.

체결반경 내에 있을 때는 두 모션블록 (1)과 (2)가 모두 활성화됩니다.

수직모션 (1)은 체결반경을 빠져나올 때에만 끝납니다.

위의 예제의 경우 체결 결과, 다음과 같은 시퀀스로 모션이 수행됩니다. 점 P2와 P4 만 "정확한 중지점"에 접근합니다.

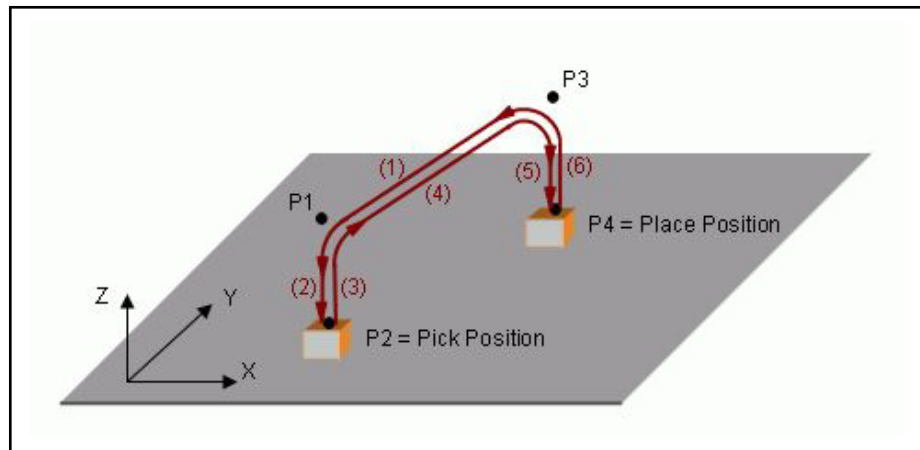


그림 7-19: 체결 모션을 사용하는 픽 앤 플레이스(Pick & Place) 애플리케이션

- 설명**
- 개체 건너편에서 시작되는 수평 모션 (1)은 체결반경에 진입하는 즉시 개체 쪽으로 내려오는 수직모션 (2)로 변경됩니다.
  - 수직모션 (2)가 개체를 잡기 위해 개체 쪽으로 구부러집니다.
  - 개체를 잡고 위쪽으로 올라가는 수직모션 (3)은 체결반경에 진입하는 즉시 수평 모션 (4)로 변경됩니다.
  - 건너편의 배치 위치로 이동하는 수평 모션 (4)는 체결반경에 진입하는 즉시 배치 위치 쪽으로 내려가는 수직모션 (5)로 변경됩니다.
  - 수직모션 (5)는 배치 위치 쪽으로 내려가 개체를 내려놓으면서 끝납니다. 새 개체가 있는지 확인합니다.
  - 새 개체가 있는 경우: 개체 없이 위쪽으로 올라가는 수직모션 (6)은 체결반경에 진입하는 즉시 개체를 가로질러 개체 쪽으로 향하는 수평 모션 (1)로 변경됩니다.
  - 새 개체가 없는 경우: 수직모션 (6)은 P3 쪽으로 이동한 후 새 개체를 대기합니다.



체결 모션을 사용하여 여러 개의 모션블럭을 차례대로 경유하여 이동하려면 일부 기본 조건이 충족되어야 합니다.

체결 모션 중에 다음 작업을 수행하면 체결이 중단됩니다.

- 동기 입력(POINT; BELT; REAL; INTEGER; BINARY)에 액세스
- 현재 위치 쿼리(POS; @POS)
- 벨트 카운터("B1")에 액세스
- 벨트 카운터(SYNC) 설정 또는 재설정
- 대기 시간(WAIT)

## 벨트 동기화

이동 벨트에서 개체를 내려놓거나 또는 개체를 이동 벨트 위에 올려 놓으려면 벨트 동기화를 사용해야 합니다.

벨트 동기화를 사용하려면 다음 조건을 충족해야 합니다.

- 키네마틱 구성에서 벨트 축이 정의되어 있어야 합니다.
- 키네마틱 구성에서 벨트 커플링 팩터가 정의되어 있어야 합니다.
- RCL 프로그램에서 벨트 카운터(채널 501 - 504)를 선언해야 합니다.
- RCL 프로그램에서 벨트 유형 4 를 정의해야 합니다(SPC\_FCT 21).
- RCL 프로그램에서 벨트 동기화 작업 영역을 정의해야 합니다(SPC\_FCT 53).



이동 벨트에서 동기화하려면 로봇이 벨트 속도까지 가속화할 수 있는 충분한 거리가 있어야 합니다. 비동기화의 경우에도 마찬가지입니다.

### SYNCHRON

벨트 동기화를 활성화하기 위한 명령어는 "SYNCHRON"이며 그 다음에 해당하는 벨트 카운터의 명칭이 따라옵니다.

동기 명령을 실행한 후 첫 번째 모션블럭에서 로봇은 벨트와 동기화됩니다. 모션블럭이 끝나면 동기화가 완료됩니다.

이 모션블럭에서 벨트 좌표에는 대상 위치에 해당하는 벨트 카운터의 값이 포함되어 있어야 합니다.



첫 번째 모션블럭의 위치가 벨트 동기화 작업 영역 앞에 있는 경우 위치가 벨트 동기화 작업 영역에 진입할 때까지 실행이 지연됩니다.



첫 번째 모션블럭이 이미 벨트 동기화 작업 영역의 마지막 가능한 시작 위치 뒤에 있는 경우에는 모션이 수행되지 않고 오류가 발생하여 프로그램이 중단됩니다.

### SYNCHRON\_END

"SYNCHRON\_END" 명령 전의 마지막 모션블럭에서 로봇은 벨트와 비동기화됩니다. 이 모션블럭에서 벨트 좌표의 값은 대상 위치 "0"이어야 합니다.

위의 예제에서는 그 결과, 다음과 같은 시퀀스로 모션이 수행됩니다.

## 첫 단계 - RobotControl

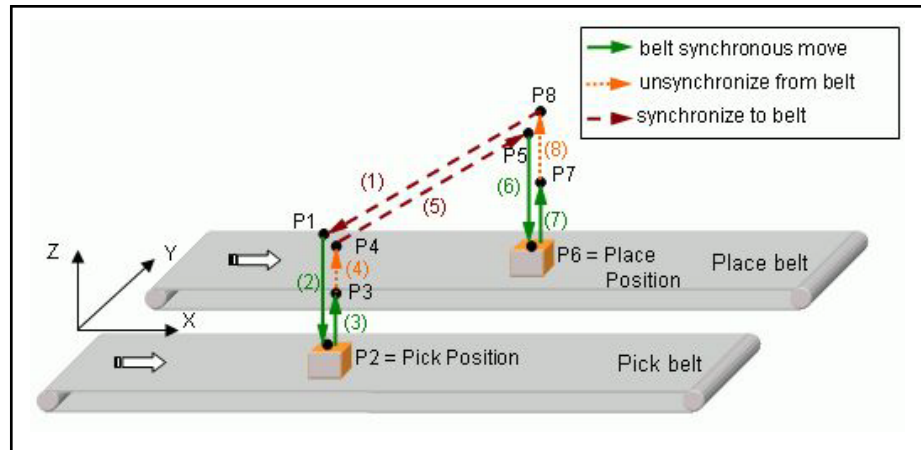


그림 7-20: 벨트 동기화를 사용하는 픽 앤 플레이스(Pick & Place) 애플리케이션 키

- P1 픽 벨트의 동기지점
- P2 픽 벨트 위 개체의 벨트 동기화 위치
- P3 P2 위의 벨트 동기화 위치
- P4 픽 벨트의 비동기 지점 = P3 쪽으로 이동하기 전의 현재 위치. Z 좌표가 위쪽을 향하도록 수정됩니다.
- P5 플레이스 벨트의 동기지점
- P6 플레이스 벨트 위 개체의 벨트 동기화 위치
- P7 P6 위의 벨트 동기화 위치
- P8 플레이스 벨트의 비동기 지점 = P7 쪽으로 이동하기 전의 현재 위치. Z 좌표가 위쪽을 향하도록 수정됩니다.

- 설명
- 동기지점 P1 쪽으로 향하는 픽 벨트의 동기모션 (1)
  - P2 다음의 벨트 동기모션 (2)
    - 개체 집어올리기
    - 현재 위치에서 비동기화 위치 P4 확인



비동기화하지 마십시오. 그렇지 않으면 그리퍼의 모션이 벨트를 기준으로 수직으로 이동하고 벨트에 있는 다음 개체가 그리퍼의 개체에 따라 이동할 수 있습니다.

- P3 다음의 벨트 동기모션 (3)
- 비동기 지점 P4 쪽으로 향하는 픽 벨트의 비동기 모션 (4)
- 동기지점 P5 쪽으로 향하는 플레이스 벨트의 동기모션 (5)
- P6 다음의 벨트 동기모션 (6)
  - 개체 내려놓기
  - 현재 위치에서 비동기화 위치 P8 확인



비동기화하지 마십시오. 그렇지 않으면 그리퍼의 모션이 벨트를 기준으로 수직으로 이동하고 판지가 있는 경우 판지가 움직일 수 있습니다.

- P7 다음의 벨트 동기모션 (7)
- 비동기 지점 P8 쪽으로 향하는 플레이스 벨트의 비동기 모션 (8)
- 새 개체를 대기한 후 (1)로 진행

## 체결 모션이 포함된 벨트 동기화

모션블럭을 체결해야 할 경우 다음과 같은 시퀀스로 모션이 수행됩니다.

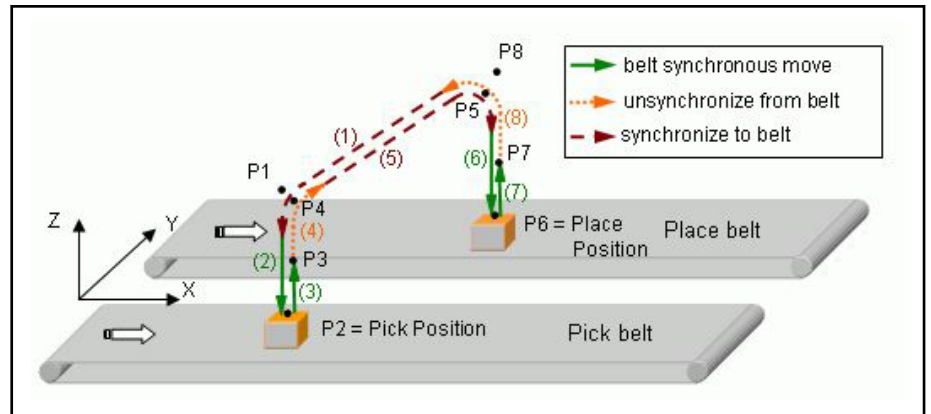


그림 7-21: 벨트 동기화 및 체결 모션을 사용하는 픽 앤 플레이스(Pick & Place)에 플리케이션

### 키

- P1 픽 벨트의 동기지점
- P2 픽 벨트 위 개체의 벨트 동기화 위치
- P3 P2 위의 벨트 동기화 위치
- P4 픽 벨트의 비동기 지점 = P3 쪽으로 이동하기 전의 현재 위치. Z 좌표가 위쪽을 향하도록 수정됩니다.
- P5 플레이스 벨트의 동기지점
- P6 플레이스 벨트 위 개체의 벨트 동기화 위치
- P7 P6 위의 벨트 동기화 위치
- P8 플레이스 벨트의 비동기 지점 = P7 쪽으로 이동하기 전의 현재 위치. Z 좌표가 위쪽을 향하도록 수정됩니다.

체결로 인해 점 P1, P4, P5 및 P8 은 직접 통과하지 않습니다.

### 설명

- 동기지점 P1 쪽으로 향하는 픽 벨트의 동기모션 (1)
  - 체결반경에 진입하면 P2 쪽으로 향하는 수직모션 (2)가 시작됩니다. 체결반경에서 빠져나올 때까지 두 모션블럭 (1)과 (2)가 모두 활성화되어 있습니다.
- P2 다음의 벨트 동기화 비체결 모션 (2)
  - 개체 집어올리기
  - P2(픽업)에서 P6(플레이스)까지의 전체 경로 확인



비동기화하지 마십시오. 그렇지 않으면 그리퍼의 모션이 벨트를 기준으로 수직으로 이동하고 벨트에 있는 다음 개체가 그리퍼의 개체에 따라 이동할 수 있습니다.

- P3 쪽으로 향하는 벨트 동기화 체결 모션 (3)
  - 체결반경에 진입하면 비동기 지점 P4 쪽으로 향하는 픽 벨트의 수직 비동기 모션 (4)가 시작됩니다. 체결반경에서 빠져나올 때까지 두 모션블럭 (3)과 (4)가 모두 활성화되어 있습니다.
- 비동기 지점 P4 쪽으로 향하는 픽 벨트의 비동기 모션 (4)
  - 체결반경에 진입하면 동기지점 P5 쪽으로 향하는 플레이스 벨트의 수평 동기모션 (5)가 시작됩니다. 체결반경에서 빠져나올 때까지 두 모션블럭 (4)과 (5)가 모두 활성화되어 있습니다.

## 첫 단계 - RobotControl

- 동기지점 P5 쪽으로 향하는 플레이스 벨트의 동기모션 (5)
  - 체결반경에 진입하면 P6 쪽으로 향하는 수직 벨트 동기모션 (6)이 시작됩니다. 체결반경에서 빠져나올 때까지 두 모션블럭 (5)와 (6)이 모두 활성화됩니다.
- P6 다음의 벨트 동기화 비체결 모션 (6)
  - 개체 내려놓기
  - 새 개체가 있는지 확인합니다.  
새 개체가 있는 경우: P6(플레이스)에서 P2(픽업)까지의 전체 경로를 확인합니다.  
새 개체가 없는 경우: 현재 위치를 기준으로 비동기화 위치 P8 까지 P6(플레이스)의 경로를 확인합니다.



비동기화하지 마십시오. 그렇지 않으면 그리퍼의 모션이 벨트를 기준으로 수직으로 이동하고 판지가 있는 경우 판지가 움직일 수 있습니다.

- P7 쪽으로 향하는 벨트 동기화 체결 모션 (7)
  - 체결반경에 진입하면 비동기 지점 P8 쪽으로 향하는 플레이스 벨트의 수직 비동기 모션 (8)이 시작됩니다. 체결반경에서 빠져나올 때까지 두 모션블럭 (7)와 (8)이 모두 활성화됩니다.
- 비동기 지점 P8 쪽으로 향하는 플레이스 벨트의 비동기 모션 (8)
 

새 개체가 없는 경우: 새 개체가 준비될 때까지 대기한 다음 P8 에서 P2(픽업)까지의 경로를 확인합니다.
- 그런 다음 (1)을 계속합니다.

## 벨트 동기화 중 체결에 대한 지침

모션이 진행 중일 때 모션을 체결하려면 다음과 같은 일부 기본 조건을 고려해야 합니다.

- 체결을 중단하는 명령:
  - 동기 입력(Point; Real; Integer; Binary)에 액세스
  - 벨트 카운터(벨트 501-504)에 액세스
  - 현재 위치 조회(POS; @POS)
  - 벨트 카운터(SYNC) 설정 또는 "변경"
  - 대기 시간(WAIT)
  - 일시 중지
- 픽업(P2)에서 플레이스(P6) 또는 그 반대 방향 사이의 모션은 체결 모션이 시작하기 전에 전체 경로가 확인된 경우에만 체결할 수 있습니다.  
이때, 픽업(P2) 또는 플레이스(P6) 위치에서 정확히 확인해야 합니다.
- 픽업(P2) 또는 플레이스(P6) 위치에서는 벨트 카운터가 모듈로 값을 초과했는지 여부를 확인해야 합니다. 그러나 벨트 카운터(SYNC 명령) 모듈로 수정은 동기 단계 중이 아닌 비동기화 P4 또는 P8 이후에만 수행할 수 있습니다.

**원인:**

벨트 카운터는 동기 입력(BELT)이므로 체결을 중단시킵니다.

**구현:**

확인을 수행할 때 모듈로 값을 초과한 경우 로컬 변수를 설정한 다음 이 변수를 사용하여 모듈로를 수정하십시오.

**제한 사항:**

## 첫 단계 - RobotControl

모듈로를 수정하려면 "SYNC" 명령을 사용해야 하지만, 이 명령도 체결을 중단시킵니다. 따라서 모듈로 값을 가능한 한 높게 설정하십시오(예: 900000, 그러나 쉼표 앞의 6 자리를 이상일 수 없습니다. 그렇지 않으면 2 밀리초 내에 벨트가 더 작은 값으로 변경될 경우 추가될 때 어떠한 영향도 미치지 않습니다.).

이러한 제한 사항을 방지하려면 항상 개체 또는 "배치할 개체"가 있을 경우에만 모듈로 수정을 수행해야 합니다.

## 예제 프로그램

다음 RCL 프로그램은 체결 모션과 벨트 동기화를 사용한 픽 앤 플레이스 (Pick&Place) 애플리케이션의 예를 보여 줍니다.

*픽 앤 플레이스(Pick&Place) - 애플리케이션*

```
;;COMPILER = RCL3
;;INT = LINEAR
;;KINEMATICS: ( 1 = Kin)
;;kin.JC_NAMES = Ax1,Ax2,Ax3,Ax4,eAx1,eAx2
;;kin.WC_NAMES = Kx,Ky,Kz,Kc,Kpk,Kpl

; *****
; Program description:
; - sample program for belt synchronous pick from infeed belt and belt synchronous
;   place on outfeed belt
; - pick and place positions are transmitted by the PLC
; - all movements between pick and place position (and vice versa) are blending
;   because: - path calculation of all blending moves is done during belt -
;             synchronous non blending movement (at point 'Pick' / 'Place')
;             - check for modulo range of belt and modulo correction of point is
;               done during belt synchronous non blending movement
;               (at point 'Pick' / 'Place')
;             - modulo correction of the belt is done during the non belt -
;               synchronous movement
; *****
PROGRAM Pi_Pl_01
; *****
; *   DECLARATION   *
; *****
VAR:
INPUT POINT :
  701 = xPickPt_i,      ; pick position at pick belt from PLC
  702 = xPlacePt_i     ; place position at place belt from PLC
INPUT REAL :
  201 = rVel_i,        ; commanded velocity from PLC
  202 = rAccel_i       ; commanded acceleration from PLC
INPUT BINARY :
  1001 = bStop_i       ; asynchronous stop signal from PLC
OUTPUT BINARY :
  1 = bFlag_q
CONST :                ; definitions
  FALSE = 0,
  TRUE = 1,
  C_BELT_NO_1 = 1,
  C_BELT_NO_2 = 2,
  C_BK_PTP = 4,        ; belt kind PTP
  MOD_VAL_B1 = 99000.0, ; modulo value for pick belt (mm)
  MOD_VAL_B2 = 99000.0 ; modulo value for place belt (mm)
BELT:
  500 + C_BELT_NO_1 = B1, ; define channel 501 as B1 (pick belt)
  500 + C_BELT_NO_2 = B2 ; define channel 502 as B2 (place belt)
POINT :
  Home,                ; world coordinate point definition
  PickApp,              ; home position
  Pick,                 ; point above point Pick to synchronize to B1
  PickDep,              ; pick position
  PickDepAsyn,          ; point above point Pick to move synchronous to
  PlaceApp,             ; point above POS to unsynchronize from B1
  Place,                ; point above point Place to synchronize to B2
  PlaceDep,             ; place position
  PlaceDepAsyn,         ; point above point Place to move synchronous to
  ; point above POS to unsynchronize from B2
REAL :
  rTotalLength,        ; variable to setup the belt area
  rBeginLength         ; variable to setup the belt area
BINARY :
  bSyncReq_B1,
  bSyncReq_B2
SPC_FCT :              ; special function to setup the belt kind
```

## 첫 단계 - RobotControl

```

21 = belt_kind(VALUE INTEGER : belt_no      ; number of belt to be configured
               VALUE INTEGER : kind_belt)  ; number of belt kind (only value 4)
SPC_FCT :      ; special function to setup the belt area
53 = belt_area(VALUE INTEGER : BeltNo      ; number of belt to be configured
               JC_POINT      : @Base       ; start pos. of working area on the belt
               REAL          : rTotalLength ; max. reachable length in belt direction
               REAL          : rBeginLength) ; latest pos. within the working area on
                                             ; the belt, where the synchronization
                                             ; move has to be started
/-----/
/ *****
/ * IMPLEMENTATION *
/ *****
/-----/
BEGIN
Init          ; call subroutine 'Init'
WAIT 5        ; wait time in seconds before proceeding with program
IF rVel_i > 0 THEN BEGIN
V = rVel_i    ; get commanded velocity from PLC
END
IF (rAccel_i > 0) THEN BEGIN
A = rAccel_i  ; get commanded acceleration from PLC
END
Pick = xPickPt_i ; get first pick position from PLC
PickApp = Pick   ; calculate intermediate point 'PickApp' (with Kpk to synchronize)
PickApp.Kz = Pick.Kz + 100 ; set z coordinate of intermediate point
LoopMain:      ; begin of main loop
PickB1        ; call subroutine 'PickB1'
PlaceB2       ; call subroutine 'PlaceB2'
If NOT bStop_i THEN BEGIN ; check for asynchronous stop signal from PLC
JUMP LoopMain ; proceed with main loop
END
MOVE TO Home   ; move to home position (point 'Home')
PROGRAM_END

/-----/
SUBROUTINE PickB1()
BEGIN
SYNCHRON B1    ; start synchronisation to pick belt with next movement
MOVE VIA PickApp ; blending move via point 'PickApp' to synchronize to pick belt
MOVE TO Pick    ; stopping move to point 'Pick' synchronous to pick belt
;----- * no blending from here * -----
WAIT 0.05
IF rVel_i > 0 THEN BEGIN
V = rVel_i    ; get commanded velocity from PLC
END
IF (rAccel_i > 0) THEN BEGIN
A = rAccel_i  ; get commanded acceleration from PLC
END
PickDep = Pick ; calculate intermediate point 'PickDep'
PickDep.Kz = Pick.Kz + 50
PickDepAsyn = POS ; calculate intermediate point 'PickDepAsyn' (from actual Pos.)
PickDepAsyn.Kz = Pick.Kz + 100
PickDepAsyn.kpk = 0 ; set belt value to zero to define synchronization position
Place = xPlacePt_i
IF B1 > MOD_VAL_B1 THEN BEGIN
bSyncReq_B1 = TRUE ; belt modulo has to be done
END
ELSE BEGIN
bSyncReq_B1 = FALSE
END
IF Place.kpl > B2 THEN BEGIN
Place.kpl = Place.kpl - MOD_VAL_B2 ; modulo correction for the place position
END
PlaceApp = Place
PlaceApp.Kz = Place.Kz + 100
PlaceDep = Place
PlaceDep.Kz = Place.Kz + 50
;----- * no blending till here * -----
MOVE VIA PickDep ; blending move via point 'PickDep' synchronous to pick belt
MOVE VIA PickDepAsyn ; blending move via point 'PickDepAsyn' to unsynchronize
SYNCHRON_END B1
IF bSyncReq_B1 = TRUE THEN BEGIN
SYNC B1 >= MOD_VAL_B1 ; belt modulo (subtraction)
END
RETURN
SUB_END

/-----/
SUBROUTINE PlaceB2()
BEGIN
SYNCHRON B2
MOVE VIA PlaceApp
MOVE TO Place

```

첫 단계 - RobotControl

```

;----- * no blending from here * -----
WAIT 0.05
IF rVel_i > 0 THEN BEGIN
    V = rVel_i ; get commanded velocity from PLC
END
IF (rAccel_i > 0) THEN BEGIN
    A = rAccel_i ; get commanded acceleration from PLC
END
PlaceDepAsyn = POS ; calculate desynchronization point 'PlaceDepAsyn' depending
; on actual position
PlaceDepAsyn.Kz = Place.Kz + 100
PlaceDepAsyn.kpl = 0
Pick = xPickPt_i ; get next pick position from PLC
IF B2 > MOD_VAL_B2 THEN BEGIN ; check if belt counter has to be modulated
    bSyncReq_B2 = TRUE ; belt modulo has to be done
END
ELSE BEGIN
    bSyncReq_B2 = FALSE
END
IF Pick.kpk > B1 THEN BEGIN
    Pick.kpk = Pick.kpk - MOD_VAL_B1 ; modulo correction for the pick position
END
PickApp = Pick ; calculate intermediate point 'PickApp' (with Kpk to synchronize)
PickApp.Kz = Pick.Kz + 100
PickDep = Pick
PickDep.Kz = Pick.Kz + 50
;----- * no blending till here * -----
MOVE VIA PlaceDep
MOVE VIA PlaceDepAsyn
SYNCHRON_END B2
IF bSyncReq_B2 = TRUE THEN BEGIN ; use 'SYNC' only if B1 > MOD_VAL_B1 otherwise it will
; avoid blending
    SYNC B2 >= MOD_VAL_B2 ; belt modulo (subtraction)
END
RETURN
SUB_END

;-----
SUBROUTINE Init()
VAR:
    POINT:
        Base
    JC_POINT: ; joint coordinate point definition
    @Base ; point to setup the startpoint of the belt area
BEGIN
    Home = (0, 0, -700, 0, 0, 0) ; define point 'Home'
    V = 1000 ; define path velocity
    A = 1000 ; define path acceleration
    R = 50 ; define maximum blending radius (mm)
    AFACTOR = 1 ; define factor for acceleration (factor for A)
    DFACTOR = 1 ; define factor for deceleration (factor for A)
    BLOCK_SLOPE ; define kind of slope for blending
    belt_kind(C_BELT_NO_1, C_BK_PTP) ; define belt kind for pick belt
    belt_kind(C_BELT_NO_2, C_BK_PTP) ; define belt kind for place belt
    rTotalLength = 600 ; define length of working area over the belt
    rBeginLength = 550 ; define latest position within the working area,
; where the synchronization movement has to be started
    Base = (-300, 0, -650, 0, 0, 0) ; set point Base in world coordinates to define base
; of the pick belt
    @Base = JC(Base) ; convert point Base to joint coordinates and copy to
; @Base
    belt_area(C_BELT_NO_1, ; execute special function to setup the working area
        @Base, ; of the pick belt
        rTotalLength,
        rBeginLength)

    Base = (-300, 100, -650, 0, 0, 0) ; set point Base in world coordinates to define base
; of the place belt
    @Base = JC(Base) ; convert point Base to joint coordinates and copy to
; @Base
    belt_area(C_BELT_NO_2, ; execute special function to setup the working area
        @Base, ; of the place belt
        rTotalLength,
        rBeginLength)

    SYNC B1, bStop_i = FALSE ; set belt counter to 0
    SYNC B2, bStop_i = FALSE ; set belt counter to 0
    MOVE TO Home ; move to home position (point 'Home')
RETURN
SUB_END

```



## 8 서비스 및 지원

당사는 고객의 필요사항을 최적으로 신속하게 지원하기 위해 글로벌 서비스 네트워크를 제공합니다. 당사의 전문가들이 고객을 돕기 위해 대기하고 있습니다. 고객은 저희에게 **24 시간/7 일** 연락할 수 있습니다.

### 독일 서비스

독일 Lohr 에 있는 기술 중심 경쟁력 센터가 전기 구동 및 제어에 대한 모든 서비스 관련 업무를 담당합니다.

서비스 **헬프데스크**와 **핫라인**에 연락하십시오.

전화: **051-260-0743, 051-260-0858**

팩스: **051-260-0865**

전자 메일: [info.brs@boschrexroth.co.kr](mailto:info.brs@boschrexroth.co.kr)

인터넷: <http://www.boschrexroth.com>

서비스, 수리(예: 배달 주소) 및 교육에 관한 추가 정보는 저희 인터넷 사이트에서 찾아볼 수 있습니다.

### 글로벌 서비스

독일 외의 국가에서는 먼저 해당 지역의 서비스 사무소에 연락하십시오. 핫라인 번호는 인터넷에서 판매 사무소 주소를 참조하십시오.

### 정보 준비

다음 정보를 준비하시면 보다 신속하고 효과적으로 도와 드릴 수 있습니다.

- 오작동 및 오작동을 일으킨 상황에 대한 자세한 설명
- 영향을 받은 제품의 유형 (특히 유형 코드 및 일련 번호)
- 당신의 연락처 (전화 및 팩스 번호, 전자메일 주소)



# 인덱스

## P

|           |    |
|-----------|----|
| PELV..... | 17 |
|-----------|----|

## L

|                         |     |
|-------------------------|-----|
| 내보내기 파일                 |     |
| 첫 단계 - FlexProfile..... | 157 |
| 첫 단계 - IndraLogic.....  | 78  |
| 첫 단계 - 모션.....          | 139 |

## C

|               |    |
|---------------|----|
| 드라이브 시스템..... | 13 |
|---------------|----|

## □

|                  |    |
|------------------|----|
| 매우 낮은 보호 전압..... | 17 |
|------------------|----|

## ㅂ

|                 |    |
|-----------------|----|
| 부적절한 사용.....    | 12 |
| 결과, 책임의 배제..... | 11 |

## ㅅ

|               |   |
|---------------|---|
| 설명서 정보.....   | 5 |
| 설명서의 유효성..... | 5 |

## ㅈ

|                          |     |
|--------------------------|-----|
| 적절한 사용                   |     |
| 사용 사례.....               | 11  |
| 소개.....                  | 11  |
| 전기 드라이브 및 컨트롤 안전 지침..... | 13  |
| 전기 드라이브 시스템.....         | 13  |
| 지원                       |     |
| 서비스 핫라인 참조.....          | 181 |

## ㅊ

|                                                 |        |
|-------------------------------------------------|--------|
| 첫 단계 - FlexProfile.....                         | 143    |
| FlexProfile 만들기.....                            | 145    |
| IndraWorks 대화 상자를 통해 FlexProfile<br>만들기.....    | 145    |
| PLC 하위 프로젝트 FlexProfile 다운로드.....               | 155    |
| PLC 를 통해 FlexProfile 만들기.....                   | 148    |
| Trace_FlexProfile.....                          | 156    |
| 기술 요구 사항.....                                   | 144    |
| 모션 하위 프로젝트 만들기.....                             | 144    |
| 모션스텝 실행, ML_FlexProfile.....                    | 151    |
| 워크플로 가상 축 → 가상 축 →<br>FlexProfile 0 → 실제 축..... | 155    |
| 프로젝트 복원.....                                    | 157    |
| 첫 단계 - IndraLogic.....                          | 23     |
| AS 편집(SFC).....                                 | 43     |
| CFC(연결구조언어)로 PlcProg 편집.....                    | 60     |
| IndraLogic 의 시각화.....                           | 66     |
| IndraWorks 시작.....                              | 24     |
| PLC 프로그램 만들기.....                               | 29     |
| PLC 프로그램의 시각화 만들기.....                          | 66     |
| POU(프로그램 유닛) 추가.....                            | 29, 34 |

|                                                    |     |
|----------------------------------------------------|-----|
| ST(구조체언어) 편집.....                                  | 41  |
| ST(구조체언어)로 PlcProg 편집.....                         | 57  |
| 기술 요구 사항.....                                      | 23  |
| 변수 값의 시각화 요소 표시.....                               | 76  |
| 복원.....                                            | 78  |
| 새 IndraWorks 프로젝트.....                             | 24  |
| 새 시각화 만들기.....                                     | 66  |
| 새 프로젝트 만들기.....                                    | 24  |
| 시각화 요소 버튼.....                                     | 74  |
| 시각화 요소 복사.....                                     | 70  |
| 시각화 요소 타원.....                                     | 67  |
| 시각화 요소에 레이블 지정.....                                | 75  |
| 시각화 테스트.....                                       | 77  |
| 시각화 편집.....                                        | 67  |
| 시뮬레이션 모드에서 테스트.....                                | 54  |
| 신호등 시스템의 컨트롤.....                                  | 29  |
| 요소 추가 및 편집.....                                    | 73  |
| 작업 구성.....                                         | 52  |
| 전역 변수 선언.....                                      | 53  |
| 제품설정.....                                          | 26  |
| 제품에 대한 기본 설정.....                                  | 26  |
| 직사각형 시각화 요소.....                                   | 72  |
| 통신 설정.....                                         | 27  |
| 평선모듈설정.....                                        | 28  |
| 평선블럭 추가.....                                       | 32  |
| 편집 FBD(평선블럭다이어그램).....                             | 35  |
| 프로그램 및 평선블럭.....                                   | 29  |
| 프로그램 및 평선블럭 편집 - 2 부.....                          | 57  |
| 프로그램 추가.....                                       | 31  |
| 프로젝트 복원.....                                       | 78  |
| 프로젝트에 새 제품 추가.....                                 | 25  |
| 첫 단계 - RobotControl.....                           | 159 |
| RoCo 프로젝트 만들기.....                                 | 160 |
| RoCoTestV**.....                                   | 169 |
| 애플리케이션 예.....                                      | 171 |
| 픽 앤 플레이스(Pick & Place) 애플리케이션.....                 | 171 |
| 픽 앤 플레이스(Pick & Place), 벨트 동기화.....                | 173 |
| 픽 앤 플레이스(Pick & Place), 체결.....                    | 171 |
| 픽 앤 플레이스(Pick & Place), 체결 모션<br>이 포함된 벨트 동기화..... | 175 |
| 필수 하드웨어 및 소프트웨어.....                               | 159 |
| 첫 단계 - 모션.....                                     | 81  |
| IndraWorks 시작.....                                 | 82  |
| PLC 프로젝트 RA1 다운로드(FBD(평선블<br>럭다이어그램)).....         | 112 |
| PLC 프로젝트 RA1 다운로드(ST(구조체<br>언어)).....              | 136 |
| PLC 프로젝트 VA1 다운로드(ST(구조체<br>언어)).....              | 123 |
| PLX 하위 프로젝트 VA1 다운로드<br>(FBD(평선블럭다이어그램)).....      | 100 |
| sercos III 드라이브 스캔.....                            | 87  |
| Trace_VA1_RA1(FBD(평선블럭다이어그<br>램)).....             | 114 |
| Trace_VA1_RA1(ST).....                             | 138 |

## 인덱스

|                                         |          |
|-----------------------------------------|----------|
| Trace_VA1(FBD(평선블럭다이어그램)).....          | 104      |
| Trace_VA1(ST).....                      | 127      |
| 가상 축 VA1 프로그래밍(FBD(평선블럭다<br>이어그램))..... | 93       |
| 가상 축 VA1 프로그래밍(ST(구조체언어))...            | 117      |
| 가상 축 추가.....                            | 88       |
| 값 쓰기, 입력 변수(FBD(평선블럭다이어<br>그램)).....    | 102      |
| 값 쓰기, 입력 변수(ST).....                    | 125      |
| 기술 요구 사항.....                           | 81       |
| 모션 스퀀스 프로그래밍(FBD(평선블럭다<br>이어그램)).....   | 91       |
| 모션 스퀀스 프로그래밍(ST(구조체언어))...              | 115      |
| 모션 하위 프로젝트 다운로드.....                    | 89       |
| 새 프로젝트 만들기.....                         | 82       |
| 실제 축 RA1 프로그래밍(FBD(평선블럭<br>다이어그램))..... | 109      |
| 실제 축 RA1 프로그래밍(ST(구조체언어))...            | 132      |
| 워크플로 가상 축 → 전자식 기어 → 실제<br>축 .....      | 113, 136 |
| 제품 설정.....                              | 84       |
| 제품에 대한 기본 설정.....                       | 84       |
| 통신 설정.....                              | 85       |
| 평선모듈 설정.....                            | 86       |
| 프로젝트 복원.....                            | 139      |
| 프로젝트에 새 제품 추가.....                      | 83       |

## 주석

Bosch Rexroth AG  
Electric Drives and Controls  
P.O. Box 13 57  
97803 Lohr, Germany  
Bgm.-Dr.-Nebel-Str. 2  
97816 Lohr, Germany  
Tel. +49 9352 18 0  
Fax +49 9352 18 8400  
[www.boschrexroth.com/electrics](http://www.boschrexroth.com/electrics)



R911341385

DOK-MLC\*\*\*-F\*STEP\*\*V12-CO01-KO-P