RB SERIES
사용자 매뉴얼

한국어
Ver. V3.9.1

Update : 2021/09

목차

머리말

제품을 설치하기 전에 본 사용자 매뉴얼을 모두 읽고 설치 과정에 따라 설명서의 지시 사항을 따라야 합니다. 본 매뉴얼의 내용은 매뉴얼이 작성된 때를 기준으로 하며, 제품에 대한 정보는 사용자에게 사전에 알리지 않고 변경될 수 있습니다.

본 매뉴얼에서 기술하고 있는 요구 조건이나 권장 사항 또는 안전 절차에 대해 확실하지 않은 부분은 ㈜ 레인보우로보틱스에 문의하시기 바랍니다. 본 매뉴얼의 일부 그림은 시스템의 개념 및 설치에 대한 이해를 돕기 위한 것이며, 실제 제품과 다를 수 있습니다.

제 1 장 제품 소개

1.1 협동로봇 시스템

㈜ 레인보우로보틱스 협동로봇 RB Series 는 누구나 쉽게 사용할 수 있는 로봇입니다. 별도의 안전장치를 사용하지 않고도 생산 효율성을 극대화할 수 있는 최상의 협업 솔루션으로써 다양한 분야에서 사람과 인접하여 제한된 공간에서 반복적인 작업을 안전하게 협업하는데 특화되어 있습니다.

- **직관적 사용성**: 그래픽 기반으로 이루어진 직관성인 UI(User Interface) 구성으로 전문가 뿐만 아니라 누구나 어렵지 않게 RB Series 을 쉽게 설정하고 운용이 가능합니다.
- **편리한 안전성**: 외부 및 자가 충돌 감지 시스템을 통해 작업중의 사고와 부상을 최소화하여 작업자에게 안전한 작업 환경을 제공합니다.
- **공간의 활용성**: RB Series 는 좁은 공간에 구애 받지 않고 모든 생산 라인에 적용될 수 있습니다. 어느 공간에서도 충분히 설치 가능하기 때문에 부담 없이 활용할 수 있습니다.

1.2 시스템 구성

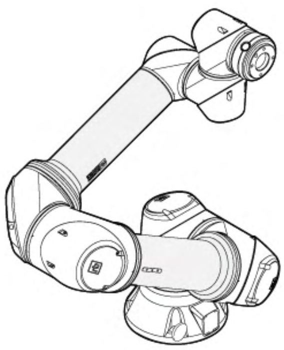
로봇 시스템 구성은 다음과 같습니다.

[스탠드 타입 컨트롤 박스 시스템 구성]

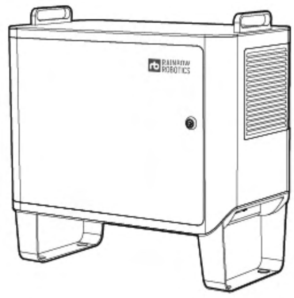
- **로봇 팔**: 소형 물체를 운반하거나 부품을 조립하는 단순 업무를 반복하는 곳에 사용할 수 있는 산업용 협동 로봇으로 다양한 로봇 손(gripper)과 여러 형태의 툴을 부착하여 사용할 수 있습니다.
- **컨트롤 박스**: 사용자가 작성한 프로그램에 따라 로봇 팔의 움직임을 제어합니다. 디지털 및 아날로그 입출력 포트를 갖추고 있어 다양한 장비 및 장치를 연결하여 사용할 수 있습니다.
- **비상정지/조그 인터페이스(스탠드 타입 컨트롤 박스 전용)**: 비상 정지 스위치가 있어서, 로봇을 운용을 정지할 수 있습니다. 재생/정지 등 간단한 프로그램 흐름 제어 버튼이 함께 있습니다.
- **태블릿 PC(옵션)**: 시스템 운영에 사용하는 장치로 로봇에게 특정 자세를 학습시키거나 로봇의 각종 제어 및 설정을 할 수 있습니다.

제공하는 시스템 구성품 및 각 구성품의 수량은 다음과 같습니다.

■ **스탠드 타입 컨트롤 박스 구성품**



[로봇 팔]



[컨트롤 박스]

[비상정지/조그 인터페이스]

[사용설명서]

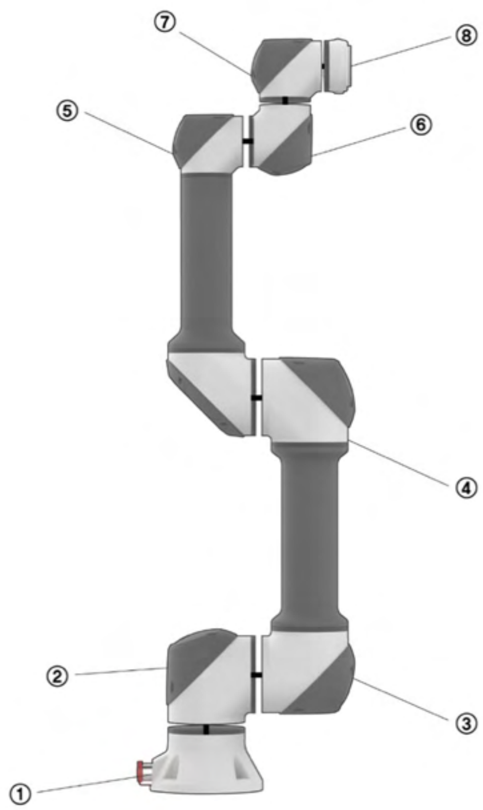
[태블릿 PC]

로봇	로봇 팔	1 EA
시스템 구성품	컨트롤 박스	1 EA
	비상정지/조그 인터페이스	1 EA
	전원 케이블	1 EA
	로봇 팔-컨트롤 박스 연결 케이블	1 EA
	옵션항목-태블릿 PC, 보호커버, USB 케이블	1 EA
기타 구성품	사용설명서(전자문서)	1 EA

※ 전원 케이블, 비상정지/조그 인터페이스 케이블, 로봇 팔 – 컨트롤 박스 연결 케이블은 제조사가 제공하는 5m 케이블을 사용할 것을 권장합니다. 실드 LAN 케이블, I/O 포트 연결 케이블, USB 케이블, 전선 통과 모델에 사용하기 위한 외부 전선 라인의 경우 3m 미만의 케이블을 사용할 것을 권장합니다.

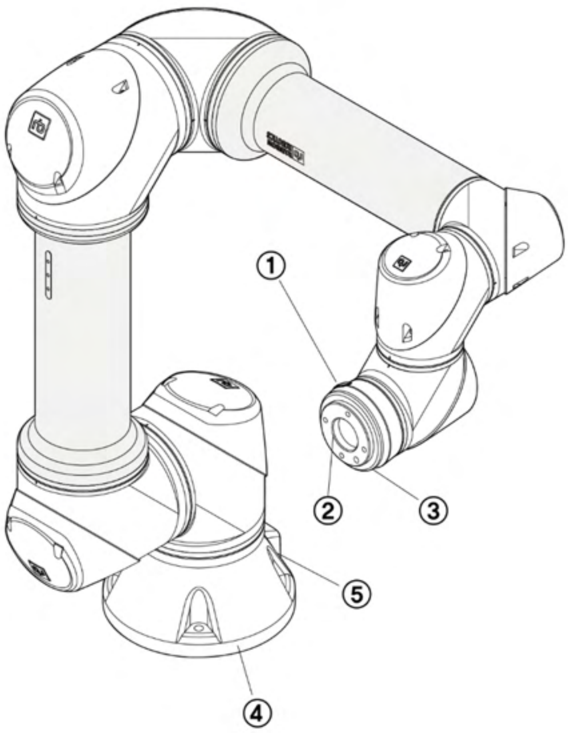
1.3 로봇 팔

■ 로봇 팔 각 부분의 명칭



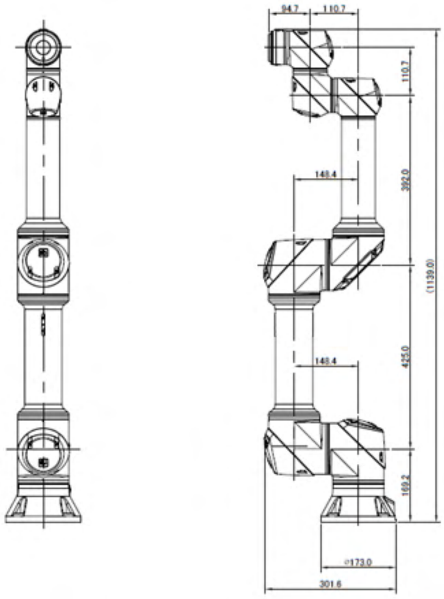
No.	명칭
①	Base
②	Base Joint
③	Shoulder Joint
④	Elbow Joint
⑤	Wrist 1 Joint
⑥	Wrist 2 Joint
⑦	Wrist 3 Joint
⑧	Tool Flange

■ 주요 부분 명칭

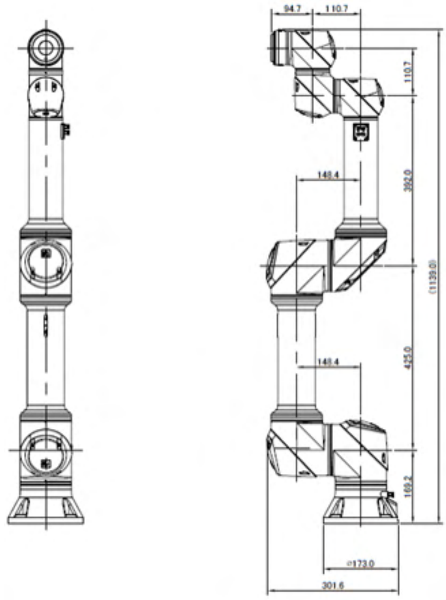


No.	명칭	설명
①	티칭 버튼	직접 교시 티칭을 위한 버튼
②	툴 플랜지	로봇에 그리퍼 또는 툴을 장착하는 부분
③	툴 I/O	그리퍼 또는 툴을 제어하기 위한 입출력 포트
④	베이스	로봇을 고정시키기 위한 부분입니다.
⑤	로봇-컨트롤 박스 커넥터	로봇 팔과 컨트롤 박스 사이의 케이블 연결 커넥터

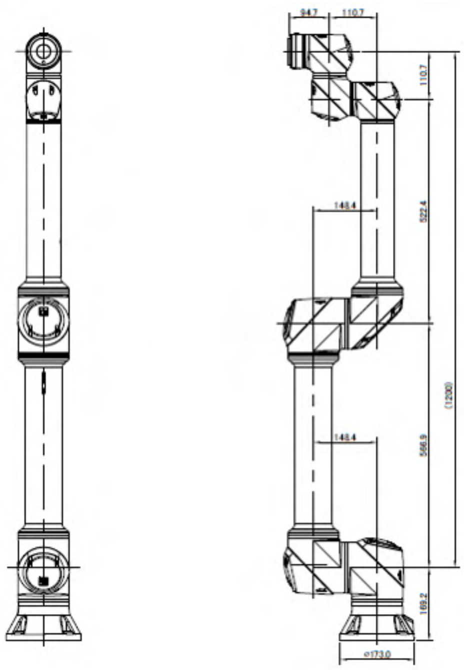
■ 로봇 팔의 형태 및 치수



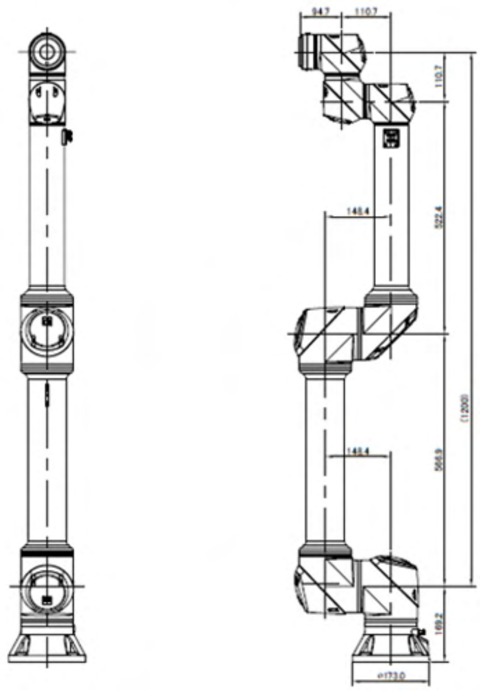
[RB5-850E]



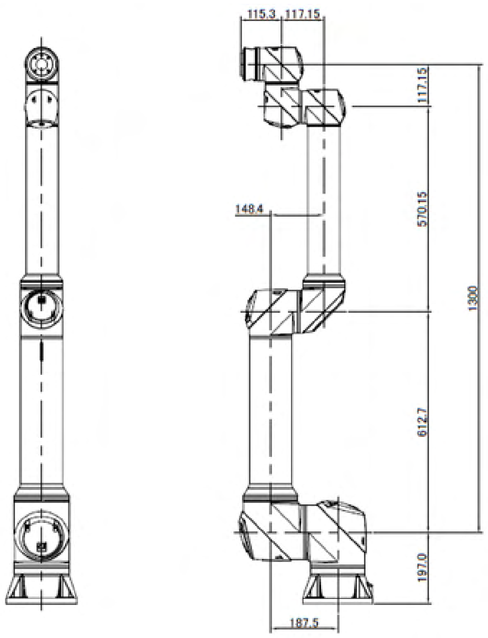
[RB5-850EA1/A2]



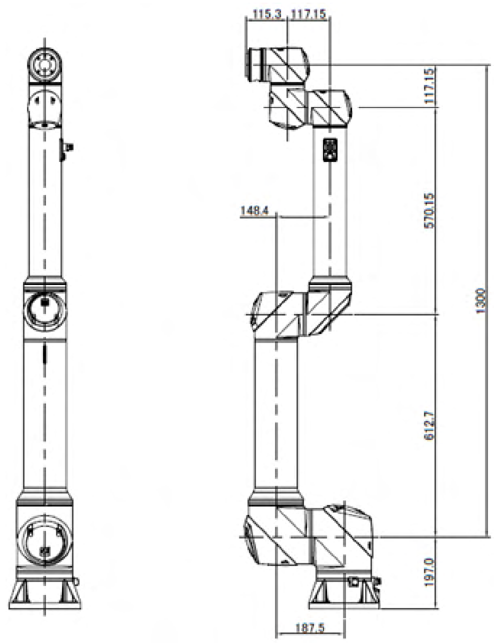
[RB3-1200E]



[RB3-1200EA1/A2]

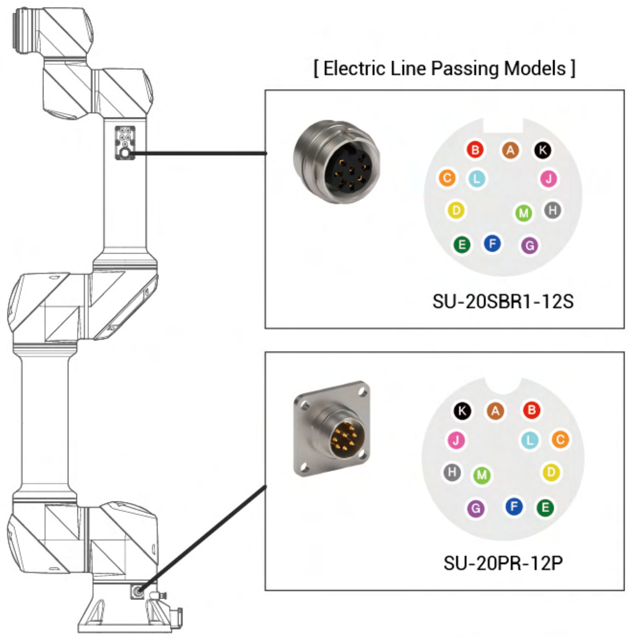


[RB10-1300E]

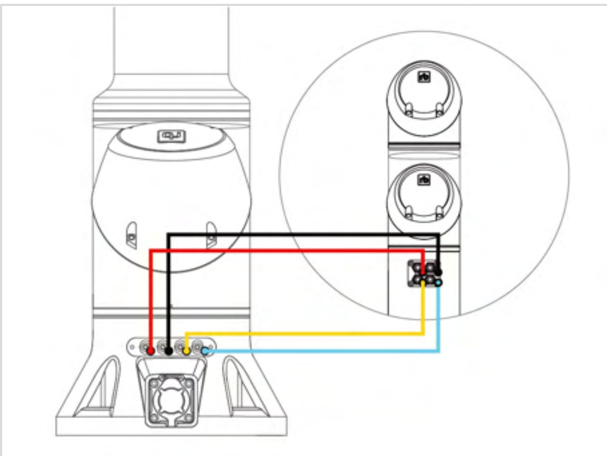


[RB10-1300EA1/A2]

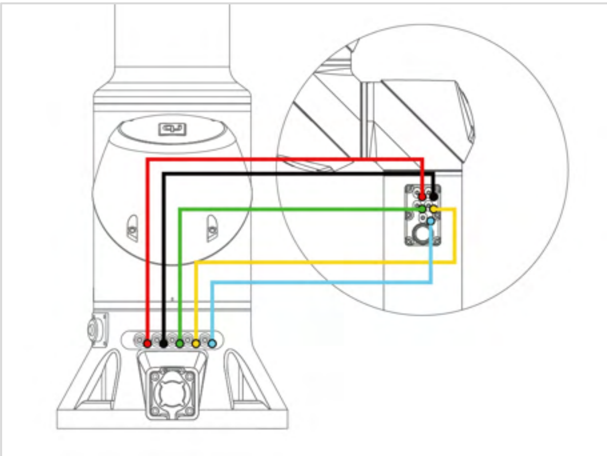
■ RB5-850EA#, RB3-1200EA#, RB10-1300EA# (공압/전선 내장형) 사용법



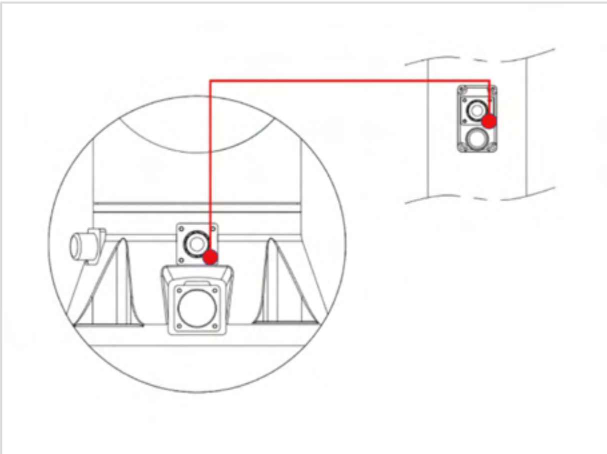
[Embedded pneumatic tubing connection]



[RB5-850EA1, RB3-1200EA1]



[RB5-850EA2, RB3-1200EA2, RB10-1300EA2]



[RB10-1300EA2]]

※ RB5-850EA#, RB3-1200EA#, RB10-1300EA# 모델의 공압 라인 및 전선 라인은 다음 표와 같이 제공되며, 위의 그림을 참조하여 사용하시기 바랍니다.

모델명	공압 라인	전선 라인
RB5-850EA1	최대 4 개(4ø 공압튜브)	없음
RB5-850EA2	최대 5 개(4ø 공압튜브)	12 핀(AWG28)
RB3-1200EA1	최대 4 개(4ø 공압튜브)	없음
RB3-1200EA2	최대 5 개(4ø 공압튜브)	12 핀(AWG28)
RB10-1300EA1	1 개(8ø 공압튜브)	없음
RB10-1300EA2	1 개(8ø 공압튜브)	12 핀(AWG28)

* 공압 라인의 갯수는 작동범위 확인 후 조정이 필요함

주의

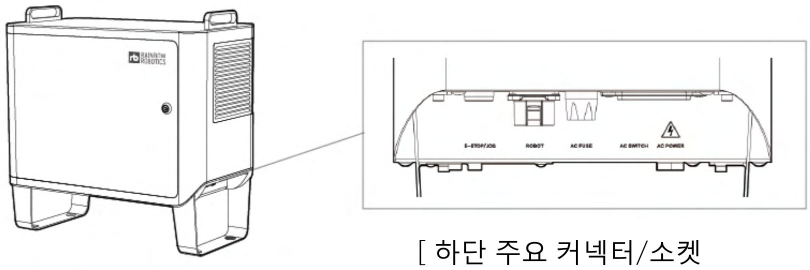
주의:

1) 공압/전선 내장형 모델의 경우, 정의된 규격 이상의 공압이나 전원을 통과시킬 경우, 하드웨어에 손상을 줄 수 있습니다.

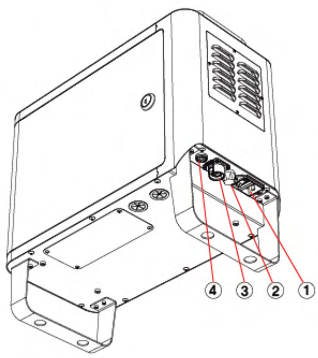
1.4 로봇 컨트롤 박스

컨트롤 박스의 전면부와 하단부 및 내부는 다음과 같습니다.

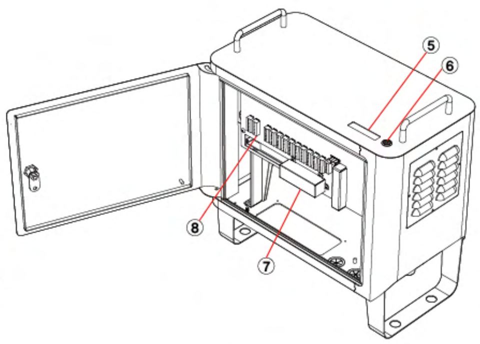
- **스탠드 타입 컨트롤 박스**



[하단 주요 커넥터/소켓]



[하단 뷰]



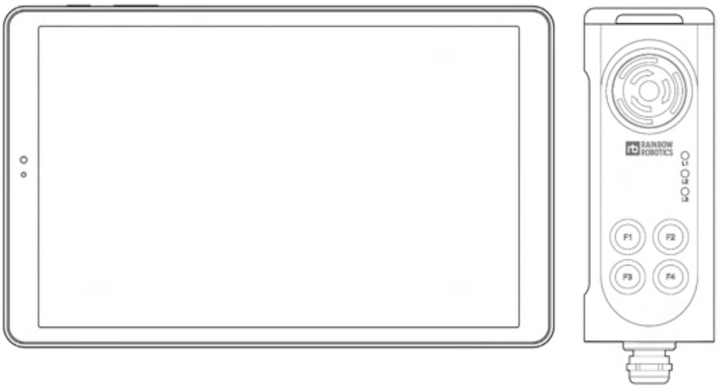
[상단 뷰]

No.	명칭
①	AC 전원 소켓 (AC POWER)
②	AC 전원 스위치 (AC SWITCH)
③	로봇 팔 케이블 커넥터 (ROBOT)
④	티칭 펜던트 커넥터 (EMO/TEACH PENDANT)
⑤	LCD
⑥	메인 스위치
⑦	USB/LAN 커넥터
⑧	I/O 포트

1.5 태블릿 PC(옵션)

태블릿 PC 는 다음과 같으며 옵션 사항이므로 기본으로 제공되지 않습니다.

■ **스탠드 타입 컨트롤 박스용**

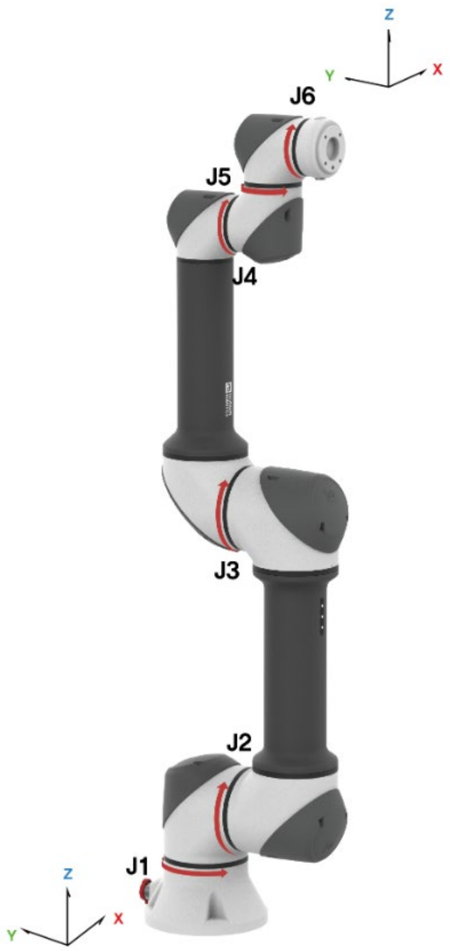


※태블릿 PC 옵션 사항을 선택하지 않을 시, 로봇 팔의 구동과 조작 그리고 티칭을 위한 전용 어플리케이션을 제공합니다 (자사 홈페이지에서 다운로드 받으실 수 있습니다).

※자사 제품과 함께 사용하기 위해서는 태블릿 설정이 요구됩니다. Appendix 를 참고하세요.

1.6 로봇의 구동 범위

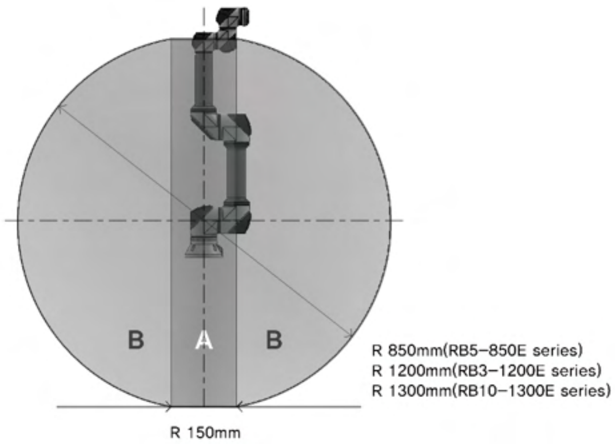
RB 로봇 팔은 6 개의 축으로 이루어져 있으며 각 관절 축의 구동 범위는 다음과 같습니다.



Joint	구동 범위
J 1	± 360 °
J 2	± 360 °
J 3	± 165 °
J 4	± 360 °
J 5	± 360 °
J 6	± 360 °

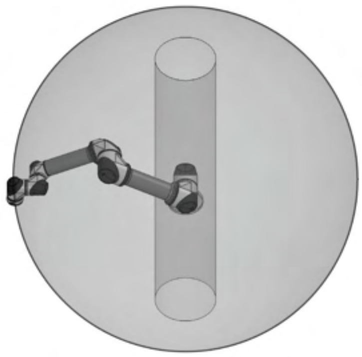
1.7 로봇의 작업영역

로봇 팔이 최대로 뻗은 상태에서 회전할 수 있는 반경은 RB5-850E Series는 850mm, RB10-1300E Series는 1300mm, RB3-1200E Series는 1200mm입니다. 로봇 베이스 바로 아래 또는 바로 윗부분(아래 그림의 A 영역과 그 주변)은 제한된 작업 영역입니다. 관절 좌표계를 사용하는 동작(예: Move J)에서는 문제가 없지만, 관성 좌표계를 사용하는 동작(예: Move L)에서는 이 영역을 통과하는 것은 제한됩니다. 이 영역은 기구적인 특이점 영역(singular area)으로써, 관성 좌표계를 사용하여 이동시, 빠른 관절 속도를 발생시키거나 정지할 수 있습니다.



A. 특이점(Singular point) 영역(직교 좌표계 움직임 시 움직임이 제한되는 영역)

B. 협동로봇이 작동 가능한 작업영역

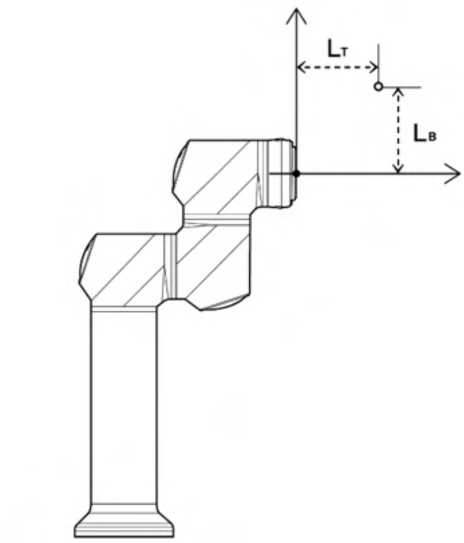


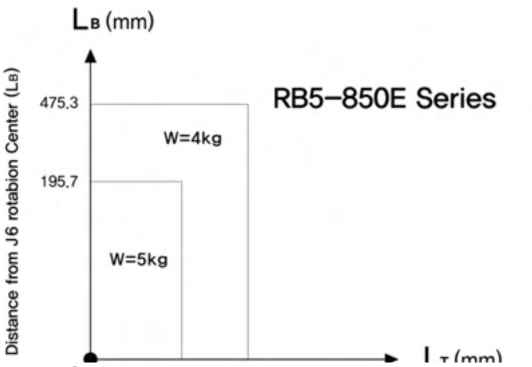
Tilt View

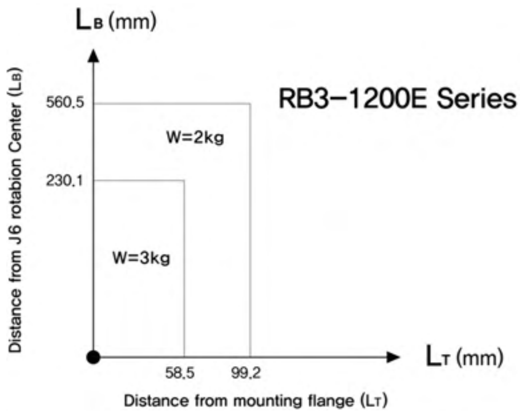
Front View

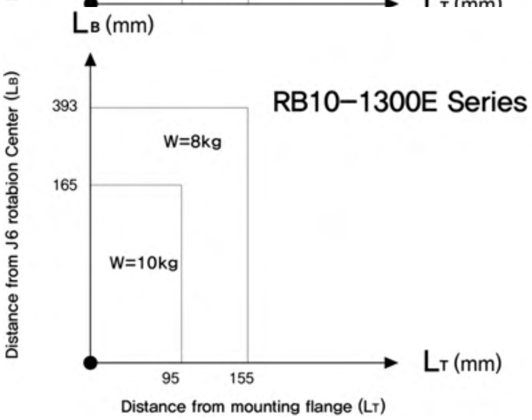
1.8 로봇 팔의 최대 가반 하중

로봇 팔의 가반 하중(payload)은 툴 플랜지와 가반 하중의 무게 중심과의 거리에 따라 다르며, 거리 별 가반 하중은 다음과 같습니다.









제 2 장 안전 및 주의사항

2.1 사용 설명서의 안전 표시

본 사용 설명서에서는 다음과 같은 안전 표시가 사용됩니다.

위험

위험:

이 표시가 있는 지시사항을 준수하지 않으면 심각한 사고를 초래할 수 있으며 그로 인해 작업자 또는 사용자가 사망하거나 심각한 상해를 입을 수 있습니다.

경고

경고:

이 표시가 있는 지시사항을 준수하지 않으면 사고가 발생할 수 있으며 그로 인해 작업자 또는 사용자가 심각한 상해를 입을 수 있습니다.

주의

주의:

이 표시가 있는 지시사항을 준수하지 않으면 제품이 손상되거나 작업자 또는 사용자가 상해를 입을 수 있습니다.

2.2 일반 안전 경고 및 유의사항

이 항목은 본 사용설명서의 다른 부분에서 반복하거나 설명할 수 있는 일반 위험과 경고 및 주의를 포함하며, 그 외 경고 및 주의도 소개되어 있습니다.

위험

위험:

1) **4 장 설치하기**의 지시 사항을 준수하여 로봇 및 전기 장비를 설치해야 합니다.

경고

경고:

1) 로봇 응용 시스템 제조자나 로봇 사용자는 본 매뉴얼의 내용을 숙지하고 운영 교육을 이수해야 합니다.
2) 로봇 팔이 자유롭게 움직일 수 있도록 충분한 공간을 확보해야 합니다.
3) 로봇을 사용할 때에는 헐거운 옷이나 장신구를 착용하지 마십시오. 머리가 길 경우에는 머리를 뒤로 묶어 로봇의 관절 등에 끼이지 않도록 주의하십시오.
4) 파손된 로봇은 절대로 사용하지 마십시오.
5) 소프트웨어에서 치명적인 오류가 발생하면 즉시 비상 정지하고 제품 공급자 또는 ㈜ 레인보우로보틱스에 연락하십시오.
6) 로봇 설치 각도, 툴 설정, 안전 설정 등의 값이 올바르게 입력되었는지 확인하십시오.
7) 일반 I/O 에는 안전 장비를 연결하지 마십시오. 안전 장비는 안전관련 I/O 에만 사용해야 합니다.
8) 티칭을 위해 태블릿 PC 를 사용할 때에는 로봇의 움직임에 주의하십시오.
9) 로봇 운영 중에는 로봇의 작동범위 안으로 들어가지 말아야 하며 로봇을 만지지 마십시오.
10) 로봇을 절대로 개조하지 마십시오. 사용자가 임의로 제품을 변경하거나 개조하여 발생한 문제에 대해서 ㈜ 레인보우로보틱스(이하 제조사)는 어떤 책임도 지지 않습니다.

11) 로봇 팔과 제어기는 장시간 사용하면 열이 발생합니다. 장시간 사용 후 로봇을 손으로 만지지 마십시오. 로봇을 만져야 하는 경우에는 제어기의 전원을 끄고 로봇을 충분히 냉각시킨 후에 만지십시오.

12) 로봇이 외부 물체와 충돌할 때에는 상당한 수준의 운동에너지가 발생하고, 이는 속도와 가반하중(payload)에 비례합니다.

13) 로봇 설치와 관련하여 권장된 설치 설정을 사용하는지 확인하십시오. 로봇 팔의 설치 방향, 툴 무게, 툴의 무게중심, 길이, 안전구성 등이 정확하게 입력되지 않으면, 직접 교시 혹은 충돌 감지 기능이 제대로 작동하지 않을 수 있습니다.

14) 직접 교시 기능은 안전한 환경에서 사용해야 합니다. 툴과 주변 설치물에 날카로운 부분이 있거나 끼임이 발생하는 부분이 있을 때에는 사용하지 마십시오.

15) 직접 교시 기능을 사용하기 전에 툴에 관련된 입력 사항(툴의 길이, 무게, 무게 중심 등)을 정확하게 입력하십시오. 실제 툴의 사양과 다른 정보를 입력하면 직접 교시 기능 사용 시 오작동을 유발합니다.

16) 직접 교시 기능 사용 시 일정 속도 이상으로 관절을 작동하면 사용자의 안전을 위해 비상 정지할 수 있습니다.

경고

경고:

1) 로봇 팔 및 컨트롤 박스에서는 작동 중에 열을 발생합니다. 로봇 팔과의 지속적인 접촉은 오작동의 원인이 될 수 있으므로, 작동 중 또는 작동 직후에 로봇 팔을 손으로 다루거나 만지지 마십시오. 로봇 팔을 조작하거나 만지기 전에 UI 화면에서 온도를 확인하거나 로봇 팔의 전원을 끄고 1 시간 이상 대기하여 충분히 냉각시킨 후 만지십시오.

주의

주의:

1) 로봇 팔에 피해를 줄 수 있는 기계 또는 다른 로봇과 함께 사용할 때에는 사용 전에 모든 기능을 따로 시험할 것을 권장합니다. 프로그래밍의 오류 또는 로봇의 장애로 다른 기계나 로봇에 손상을 입히거나 입을 경우 제조사는 책임을 지지 않습니다.
2) 로봇 팔을 강한 자기장에 노출시키지 마십시오. 로봇이 손상될 수 있습니다.

경고:

1) 전기 장치로 인한 감전위험이 있는 곳에 경고표지를 부착합니다.
2) 표지를 찢거나, 손상시키거나, 제거하지 마십시오. 표지가 부착된 부품이나 장치는 물론 주변의 부품을 다룰 때 조심하십시오.
3) 전기적인 충격을 피하려면 내부의 전기 부품을 만지지 마십시오.

2.3 사용 용도

본 로봇 팔은 툴을 이용한 물체의 이송, 조립 등에 사용할 수 있으며 사양에 명시된 환경에서만 사용할 수 있습니다. 안전 기능을 탑재하여 물리적 보호 펜스 없이 협동 작업이 가능하지만 로봇을 적용한 최종 시스템의 경우 전체 시스템에 대한 위험성 평가를 수행한 후 사용해야 합니다. 아래와 같은 용도 및 환경에서 사용하는 것은 부적절한 사용으로 간주하며 제조사는 이로 인하여 발생하는 로봇의 손상 또는 간접적 손해 또는 우발적, 특수, 결과적 손해에 대하여 책임지지 않습니다.

- 폭발의 가능성이 있는 환경에서의 사용
- 의료 및 인명과 관계된 용도
- 사람 및 동물의 이송 용도
- 위험성 평가 없이 사용
- 안전 기능의 성능이 불충분한 곳에서의 사용
- 성능/환경 사양을 벗어난 사용

※ 위의 항목으로만 제한되지 않습니다.

2.4 잠재적 위험

로봇 팔을 적용한 최종 시스템이 로봇 팔에 내장된 안전 기능만으로 위험을 충분히 감소시킬 수 없다는 결과가 나오면 추가적인 보호 대책을 수립해야 합니다. 다음과 같은 잠재적인 위험을 고려해야 합니다.

- 로봇 관절 부위에 손가락 등이 끼어 발생할 수 있는 상해(협착)
- 툴의 날카로운 면이나 모서리에 의한 상해(찔림, 관통)
- 로봇 주변에 위치한 물체에 의한 상해(찔림, 관통, 넘어짐)
- 독성 및 유해 물질에 대한 작업 시 발생할 수 있는 상해(피부 손상, 호흡곤란)
- 로봇과의 충돌로 인한 상해(멍듦, 골절)
- 체결 부위가 완전히 고정되지 않아 발생할 수 있는 상해
- 툴에서 분리되거나 낙하한 물체로 인한 상해

※ 최종 시스템에 따라 발생 가능한 잠재적 위험은 다를 수 있습니다.

2.5 책임의 한계

본 사용 설명서는 안전에 영향을 주는 모든 주변 장치에 대해 다루지 않습니다. 시스템 설치자는 로봇 팔이 설치되는 국가의 안전 규정 및 법률에 의거해 안전 요건을 준수해야 합니다. 로봇 팔은 주변장치를 연동한 최종 시스템으로 구성되며 본 사용설명서는 최종 시스템의 설계, 설치, 운영 및 안전장치를 포함한 모든 주변 장치에 대해 다루지 않습니다. 로봇 팔을 적용한 최종 시스템은 설치되는 국가의 규정 및 법률에 부합하도록 안전 요구 사항에 맞추어 설계되고 설치되어야 합니다.

로봇 팔을 적용한 최종 시스템을 구현하는 주체나 사용자는 아래에 대한 책임을 가집니다.

- 최종 시스템에 대한 위험도 평가
- 위험도 평가의 결과에 따른 안전 장치의 추가
- 시스템이 올바르게 설계, 설정, 설치되었는지 확인
- 시스템에 대한 사용법 정의
- 사용 및 안전에 관한 중요 표시나 연락처의 명시
- 사용 설명서 등의 기술 문서 제공

※ 위의 항목으로만 제한되지 않습니다. 본 설명서의 안전 사항을 준수하는 것이 발생할 수 있는 모든 위험을 방지할 수 있다는 것을 의미하지는 않습니다.

2.6 운송 및 운반

운송 및 운반 시 최소 2 인 이상이 필요하며, 제품이 떨어져 파손되지 않도록 주의하십시오. 운송 및 운반 중에 제품이 파손될 경우에 품질 보증에서 제외됩니다.

경고

경고:

1) 운송 및 운반 중에 제품이 떨어져 파손되지 않도록 주의하십시오. 이로 인해 제품이 파손될 경우에 품질 보증에서 제외됩니다.
2) 로봇 팔의 운송 시, 강한 진동 또는 충격이 발생하면 시스템이 손상될 수 있습니다. 제조사에서 제공한 포장 상자를 이용하여 로봇을 운송해야 합니다.

2.7 비상 정지

비상 시 로봇 팔을 긴급히 정지시키기 위해 로봇 컨트롤 박스와 연결되어 있는 비상 정지 스위치를 이용하여 로봇을 정지시킬 수 있습니다.

다음은 스탠드 타입 컨트롤 박스의 비상 정지 스위치 작동 방법입니다.

■ **비상 정지하기**

비상 정지 버튼을 눌러 로봇 팔을 즉시 정지할 수 있습니다.

■ **비상 정지 해제하기**

비상 정지 버튼을 시계 방향으로 돌리면 비상 정지가 해제됩니다.

2.8 사용자 안전

사용자의 안전을 위해 다음의 내용들을 주의하기 바랍니다.

- **로봇 무동력 조작**

비상 상황에서 로봇 팔에 전원을 인가할 수 없거나 인가해서 안되는 경우, 강제로 관절에 힘을 가해 로봇 팔을 움직일 수 있습니다(강제 백드라이빙). 강제 백드라이빙을 수행하여 관절을 움직이기 위해서는 로봇 팔을 힘껏 밀거나 당겨야 합니다.

주의

주의:

1) 무동력 상태에서 과도한 힘을 사용하여 관절을 움직이면 구동부에 무리가 갈 수 있으니 주의하십시오. 과도한 힘으로 인한 고장은 제조사가 책임지지 않습니다.

2.9 안전 제어기

레인보우로보틱스 로봇 모델 RB5-850E Series, RB3-1200E Series, RB10-1300E Series 의 Safety Control System 은 ISO 13849-1 Cat3. PLd 를 만족합니다.

2.10 위험성 평가

위험성 평가는 로봇 팔이 구성되어 있는 시스템을 구성할 때 중요한 요소입니다. 로봇 설치에 대한 안전은 로봇 팔을 전체 시스템에 통합하는 방식에 따라 달라지기 때문에 로봇 자체 만으로는 위험성 평가를 할 수 없습니다. 로봇의 위험성 평가를 진행하기 위해 로봇이 포함된 시스템을 구성하는 사용자는 ISO 12100 및 ISO 10218-2 의 지침에 따라 로봇을 서치하고 운영해야 하며, 추가적으로 기술규격서 ISO/TS 15066 을 참조하십시오.

위험성 평가는 반드시 로봇 팔을 설치한 직후에 수행해야 하며, 주변 환경 및 로봇 설정에 적합한 안전 구성 설정, 추가 비상 정지 버튼 및 보호 수단의 필요성을 알아내는 것이 위험성 평가의 주된 내용입니다.

협동 로봇의 안전 관련 기능은 안전구성 설정 메뉴에서 구성이 가능하며, 다음 기능을 제공합니다.

- **I/O 설정:** 출력 단자에 안전과 관련된 중요한 정보를 출력하기 위해 설정할 수 있습니다.
- **속도 조절:** 로봇 팔의 이동 속도를 조절할 수 있습니다.
- **충돌감지 감도 조절:** 로봇이 주변 물체와 충돌하게 되면 자동으로 멈추는데 이때 충돌을 감지하는 민감도를 조절할 수 있습니다.

위의 안전관련 기능으로 위험이 충분하게 감소하거나, 제거되지 않는다면, 추가 보호 장치를 더하여 위험성을 제거하도록 합니다. 설치와 관련하여 국제 표준 및 국내 법규의 관련 요건을 따르지 않거나, 상기의 위험성 평가에서 검토하지 않아 발생하는 사고는 제조사가 책임지지 않습니다.

제 3 장 안전 기능

3.1 소개

이 장에는 로봇의 전원을 켜기 전에 협동로봇 RB Series 시스템 통합자 및 사용자가 읽고 이해해야 하는 중요한 안전 정보가 소개되어 있습니다.

RB Series 는 다양한 안전 기능 및 안전 장치 인터페이스를 제공하여 사용자와 장치를 보호할 수 있습니다. 안전 기능 및 인터페이스는 ISO 13849-1 에서 설명하는 Category 3, Performance Level d (PL d)와 IEC 62061 에서 설명하는 Hardware Fault Tolerance 1, Safety Integrity Level 2(SIL 2)를 만족합니다.

주의

주의:

1) 로봇이 설치되는 경우에 따라 시스템 통합자는 위험성 평가를 수행해야 하며, 그에 따라 안전 감시 기능 및 인터페이스를 사용하여 작업 공간을 구성해야 한다.
2) 로봇의 안전 기능, 인터페이스에서 결함을 발견하면 Stop Category 0 가 개시됩니다.
3) 결함의 예시는 비상 정지 회로의 케이블 파손, 추가 안전 장치의 잘못된 결선, 추가 안전 장치의 중복되지 않은 결선 방식(본 장의 5 절 참조) 등이 있습니다.
4) 시스템 통합자 및 운용자는 로봇이 내부적으로 수행하는 안전 감시 기능이 있다는 사실을 인지하고 있어야 하며, 안전 감지 기능은 작업 수행 중에 비상정지 스위치의 동작, 보호 정지 장치의 동작뿐만 아니라, 로봇 팔의 위치, 속도, 운동량 등 물리적 제한치 이상의 동작에 대해서도 지정된 정지 모드로 로봇을 정지할 수 있습니다 (안전 감시 기능은 본 장의 3 절 참조).
5) 시스템 통합자 및 운용자는 위에 설명한 오류 및 안전 감시 기능의 동작으로 로봇이 정지하기까지의 시간과 정지거리가 발생하는 것을 고려해야 합니다. 시스템 통합자는 정지 거리 및 시간(본 장의 4 절 참조)을 고려하여 위험성 평가를 진행해야 합니다.
6) 시스템 통합자 및 운용자는 로봇의 관절, 로봇/TCP 의 움직임을 제한하기 위한 안전 감시 기능이 있는 사실을 인지하며 로봇의 동작 범위를 선정해야 합니다. TCP 는 로봇 팔 끝 중심점에 오프셋이 더해진 위치를 의미합니다.

위험

위험:

1) 시스템 통합자는 로봇 전원 인가 전 위험성 평가를 진행해야 하며, 위험성 평가에 의해 결정된 것과 다르게 사용하거나 파라메터를 지정하면 충분히 감소되지 않은 위험이 발생할 수 있습니다.
2) 추가 안전 장치의 결선 시 로봇과 제어반의 전원이 모두 차단되어 있어야 합니다.
3) 추가 안전 장치의 장착 시 기구적으로 사용 시 문제가 없도록 조치를 해야 합니다. 예시로 라이트 커튼의 사용 시 바닥/고정대와 단단히 고정되어 있어야 하며, 로봇 운용 중에 이동, 진동이 발생되지 않도록 해야 합니다.
4) 안전 기능 인터페이스는 모두 24V 로 지정되어 있습니다. 다른 전압의 장치를 연결할 경우 장비 손상 및 화재의 원인이 될 수 있으므로 주의해야 합니다.
5) Tool Flange 에 장착되는 장치의 신호는 안전 기능에 포함되지 않습니다. 안전 장치를 Tool Flange 케이블에 연결하여 사용하지 마십시오.

3.2 정지 카테고리

안전 기능을 통해 로봇은 IEC 60204-1 에 따라 정의된 세 가지 유형의 정지 카테고리를 시작합니다.

정지 카테고리	설명
0 [STO]	즉시 로봇의 전원을 차단하여 정지합니다. *관절 브레이크 마모가 발생하여 로봇의 수명이 단축될 수 있습니다. 불가피한 경우가 아니면 사용을 지양합니다. **로봇의 전원이 차단된 상태이므로 위험 제거 후 다시 사용 시 활성화 작업이 필요합니다.
1 [SS1]	로봇의 모든 관절을 최대로 감속하여 멈춘 후 전원을 차단하여 정지합니다. *로봇의 전원이 차단된 상태이므로 위험 제거 후 다시 사용 시 활성화 작업이 필요합니다.
2 [SS2]	로봇의 모든 관절을 최대로 감속하여 멈춘 후 SOS 상태로 진입합니다. *SOS: 로봇 전원이 인가되고 활성화가 되어 있는 상태로 현재 위치를 유지, 위치 변화가 감지될 경우 Stop Category 0 개시. ** 전원이 차단되지 않으므로 위험 제거 후 즉시 사용이 가능합니다.

주의

주의:

1) ISO10218-1 5.5.2 와 5.5.3 에 따라 비상정지, 보호 정지에 적합한 정지 카테고리를 선정해야 합니다.
2) 비상정지의 경우 정지 카테고리 0, 1 중에서 선정해야 합니다.
3) 비상정지가 발생된 경우 활성화 작업이 필요합니다.
4) 보호정지의 경우 최소한 하나는 정지 카테고리 0, 1 중에서 선정해야 합니다.
5) 추가 보호정지는 정지 카테고리 2 를 사용할 수 있습니다.

3.3 기능 안전

제조업체는 설치 위치에 대해 다음 조건을 충족할 것을 권장합니다. 협동로봇 RB Series 의 안전 기능은 위험성 평가에 의해 결정된 로봇 시스템의 위험을 줄이기 위해 사용됩니다.

안전기능의 파라미터는 초기 출하 시 설정되어 있으며, 시스템 통합자는 위험성 평가에 따라 일부 항목을 변경할 수 있습니다. 위치와 속도 항목은 로봇의 Base 를 기준으로 합니다.

다음은 RB Series 가 제공하는 안전 기능 사양입니다.

	안전 기능	PL & Category
안전 정지 기능	SF.1 STO (Safe Torque Off)	PL d, Category 3
	SF.2 SS1 (Safe Stop 1)	PL d, Category 3
	SF.3 SS2 (Safe Stop 2)	PL d, Category 3
안전 모니터링 기능	SF.4 SOS (Safe Operating Stop)	PL d, Category 3
	SF.5 SLP (Safely-Limited Position)	PL d, Category 3
	SF.6 SLS (Safely-Limited Speed)	PL d, Category 3
	SF.7 SLA (Safely-Limited Acceleration)	PL d, Category 3
	SF.8 SLI (Safely-Limited Increment)	PL d, Category 3
	SF.9 SLT (Safely-Limited Torque)	PL d, Category 3
	SF.10 RPL (Robot Position Limit)	PL d, Category 3
	SF.11 TSL (TCP Speed Limit)	PL d, Category 3
	SF.12 CBPL (Control Box Power Limit)	PL d, Category 3
비상 정지	SF.13 EMS1 (Emergency Stop1)	PL d, Category 3
	SF.14 EMS2 (Emergency Stop2)	PL d, Category 3
보호 정지	SF.15 PRS (Protective Stop)	PL d, Category 3
	SF.16 HSS (Hard Safeguard Stop)	PL d, Category 3
	SF.17 SSS (Soft Safeguard Stop)	PL d, Category 3

- **STO(Safe Torque Off)**: 이 기능은 힘을 발생시키는 동력이 모터에 공급되는 것을 방지합니다. 회전을 유발할 수 있는 전원은 모터에 공급되지 않습니다. 이 안전 관련 하부 기능은 IEC 60204-1 의 정지 카테고리 0 에 따른 제어되지 않은 정지에 해당합니다.

- **SS1(Safe Stop 1)**: 이 기능은 다음 중 하나로 지정됩니다. a) SS1-d (Safe-Stop 1 감속 제어)는 모터를 정지하기 위해 선택한 한계 내에서 모터 감속 속도를 시작 및 제어하고 모터 속도가 지정된 한계 미만일

때 STO 기능을 수행합니다. 또는 b) SS1-r (안전 정지 1 램프 모니터링)는 모터를 정지하기 위해 선택한 한계 내에서 모터 감속 속도를 시작 및 모니터링하고 모터 속도가 지정된 한계 미만일 때 STO 기능을 수행합니다. 또는 c) SS1-t (안전 정지 1 시간 제어)는 모터 감속을 시작하고 애플리케이션 특정 시간 지연 후 STO 기능을 수행합니다. 이 안전 관련 하부 기능은 IEC 60204-1 의 정지 카테고리 1 에 따라 제어된 정지에 해당합니다. 우리 시스템은 SS1-t 를 사용합니다.

- **SS2(Safe Stop 2)**: 이 기능은 a) SS2-d (안전 정지 2 감속 제어)가 선택한 한계 내에서 모터 감속 속도를 시작 및 제어하여 모터를 정지하고 모터 속도가 지정된 한계 미만일 때 안전 작동 정지 기능을 수행합니다. 또는 b) SS2-r (안전 정지 2 램프 모니터링)는 모터를 정지하기 위해 선택한 한계 내에서 모터 감속 속도를 시작 및 모니터링하고 모터 속도가 지정된 한계 미만일 때 안전 작동 정지 기능을 수행합니다. 또는 c) SS2-t (안전 정지 2 시간 제어)는 모터 감속을 시작하고 애플리케이션 특정 시간 지연 후 안전 작동 정지 기능을 수행합니다. 이 안전 관련 하부 기능 SS2 는 IEC 60204-1 의 정지 카테고리 2 에 따라 제어된 정지에 해당합니다. 우리 시스템은 SS2-t 를 사용합니다.

- **SOS(Safe Operating Stop)**: 이 기능은 모터가 정지 위치에서 정의된 양 이상 벗어나는 것을 방지합니다. PDS (SR)는 모터에 에너지를 제공하여 외부 힘에 저항할 수 있도록 합니다. 작동 중지 기능에 대한이 설명은 외부 (예: 기계) 브레이크 없이 PDS (SR)를 통한 구현을 기반으로 합니다.

- **SLP(Safely-Limited Position)**: 이 기능은 모터 shaft (또는 리니어 모터를 사용하는 경우 이동기)가 지정된 위치 제한을 초과하는 것을 방지합니다.

- **SLS(Safely-Limited Speed)**: 이 기능은 모터가 지정된 속도 제한을 초과하는 것을 방지합니다.

- **SLA(Safely-Limited Acceleration)**: 이 기능은 모터가 지정된 가속 또는 감속 한계를 초과하는 것을 방지합니다.

- **SLI(Safely-Limited Increment)**: 이 기능은 모터 축이 지정된 시간 내에 지정된 위치 증가 한계를 초과하는 것을 방지합니다.

- **SLT(Safely-Limited Torque)**: 이 기능은 모터가 지정된 토크 (또는 리니어 모터를 사용하는 경우 힘) 제한을 초과하는 것을 방지합니다.

- **RPL(Robot Position Limit)**: 이 기능은 로봇 팔의 TCP (공구 중심점) 또는 몸체 프레임이 지정된 공간 영역을 초과하는 것을 방지합니다.

- **TSL(TCP Speed Limit)**: 이 기능은 로봇 팔의 TCP 속도가 지정된 속도를 초과하는 것을 방지합니다.

- CBPL(Control Box Power Limit): 이 기능은 컨트롤 박스의 전력 소비가 지정된 제한을 초과하는 것을 방지합니다.

- **EMS1(Emergency Stop1)**: 이 기능은 티칭 펜던트 장치의 비상 정지 스위치가 활성화될 때 정지 모드를 활성화합니다. 정지 모드는 SF.2 입니다.

- **EMS2(Emergency Stop2)**: 이 기능은 컨트롤 박스의 특수 I/O 포트가 활성화되면 정지 모드를 활성화합니다. 이러한 포트는 사용자가 자신의 스위치 장치를 연결할 수 있도록 제공됩니다. 정지 모드는 SF.2 입니다.

- **PRS(Protective Stop)**: 이 기능은 컨트롤 박스의 특수 I/O 포트가 활성화되면 정지 모드를 활성화합니다. 이러한 포트는 사용자가 자신의 보호 장치를 연결할 수 있도록 제공됩니다. 정지 모드는 SF.2 입니다.

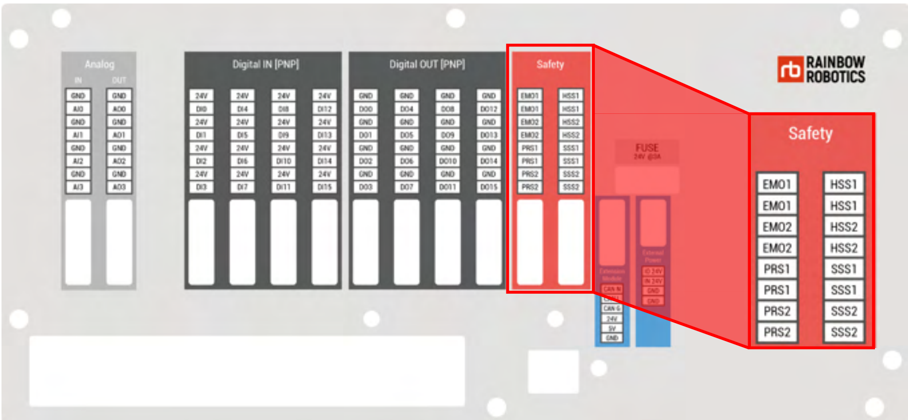
- **HSS(Hard Safeguard Stop)**: 이 기능은 컨트롤 박스의 특수 I/O 포트가 활성화되면 정지 모드를 활성화합니다. 이러한 포트는 사용자가 자신의 보호 장치를 연결할 수 있도록 제공됩니다. 정지 모드는 SF.1 입니다.

- **SSS(Soft Safeguard Stop)**: 이 기능은 컨트롤 박스의 특수 I/O 포트가 활성화되면 정지 모드를 활성화합니다. 이러한 포트는 사용자가 자신의 보호 장치를 연결할 수 있도록 제공됩니다. 정지 모드는 SF.3 입니다.

3.4 안전 장치 장착 위치

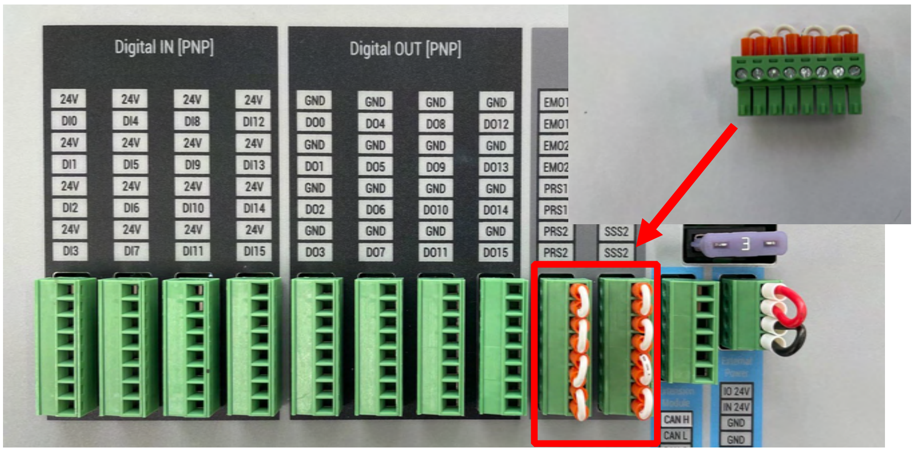
협동로봇 RB Series 는 기본으로 제공되는 비상정지 스위치 외에 시스템 통합자가 위험성 평가를 통하여 요구되는 안전 장치를 추가로 장착할 수 있습니다.

Safety 전용 접점 단자는 16 개의 포트로 구성되어 있습니다. 이 단자는 이중화된 전용 접점 입력 단자입니다.



추가 포트는 4 개의 장비를 장착하여 사용할 수 있으며, 추가 안전 장비 없이 작동할 수 있는 상태로 출고됩니다.

외부 안전 장치를 연결하지 않고 사용하는 경우에는 아래와 같이 기본 제공되는 접점 입력을 연결하여 사용하세요.



안전 장치 장착 포트 사양은 다음과 같습니다.

- **EMO**

위험성 평가를 통한 여분의 비상정지 스위치 장착이 필요한 경우 사용하는 포트입니다.

비상정지 스위치는 IEC 60947-5-5 를 준수하는 제품을 사용해야 합니다.

EMO 를 통해 발생되는 비상정지는 정지 카테고리 1 로 지정되어 있습니다.

- **PRS**

위험성 평가를 통한 한 개 이상의 보호정지 장치를 연결할 때 사용하는 포트입니다.

보호 정지 장치는 ISO 10218-2 의 5.3.8.3 을 준수한 제품을 사용해야 합니다.

PRS 를 통해 발생되는 보호정지는 정지 카테고리 1 로 지정되어 있습니다.

- **HSS**

위험성 평가를 통한 한 개 이상의 보호정지 장치를 연결할 때 사용하는 포트입니다.

보호 정지 장치는 ISO 10218-2 의 5.3.8.3 을 준수한 제품을 사용해야 합니다.

HSS 를 통해 발생되는 보호정지는 정지 카테고리 0 으로 지정되어 있습니다.

- **SSS**

위험성 평가를 통한 한 개 이상의 보호정지 장치를 연결할 때 사용하는 포트입니다.

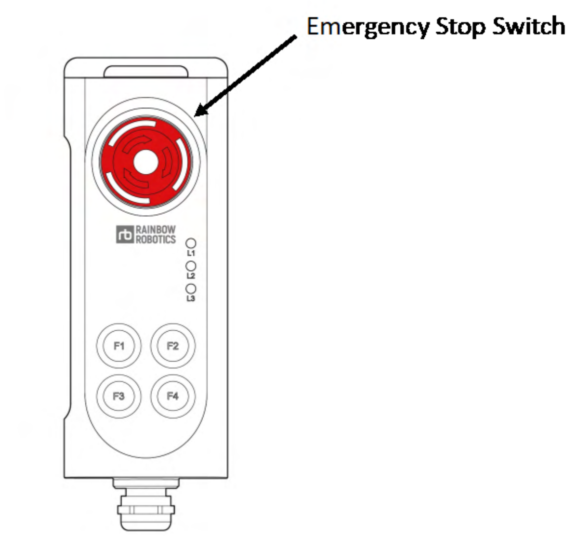
보호 정지 장치는 ISO 10218-2 의 5.3.8.3 을 준수한 제품을 사용해야 합니다.

SSS 를 통해 발생되는 보호정지는 정지 카테고리 2 로 지정되어 있습니다.

3.5 비상정지 스위치

협동로봇 RB Series 는 작업자는 비상 상황에 대비하여 로봇을 정지시킬 수 있는 비상정지 스위치 버튼을 사용할 수 있습니다.

비상 상황이 발생할 경우 펜던트 상단에 있는 비상정지 스위치 버튼을 눌러 로봇을 즉시 정지해야 합니다.



주의

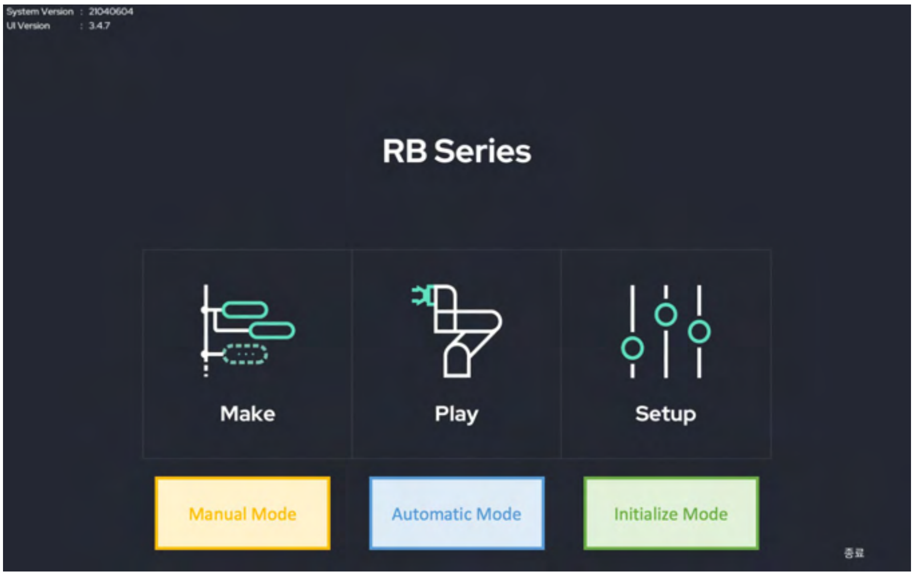
주의:

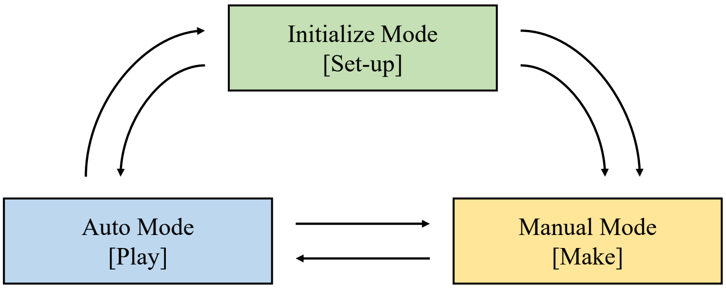
1) 비상정지 스위치 버튼의 경우 정지 카테고리 1 로 지정되어 있습니다.
2) 비상정지 스위치 버튼을 시계 방향으로 돌리면 비상정지 기능을 해제할 수 있습니다.
3) 추가 비상정지 스위치 버튼이 필요한 경우에 컨트롤 박스를 통해 사용할 수 있습니다.
4) 비상정지를 위험 감소 방법으로 사용해서는 안 되고, 2 차 보호 장치로 사용해야 합니다.

3.6 동작 모드

협동로봇 RB Series 의 동작 모드는 아래와 같이 구성되어 있습니다.

자동 모드 진입 시 비밀번호를 통해 접근해야 합니다.





주의

주의:

1) 출고 시 자동모드 진입 비밀번호는 설정되어 있지 않습니다. 다른 누군가가 접근하지 않도록 비밀번호를 설정하고 사용해야 합니다.
2) Auto Mode 에 진입 전 사용자는 주변에 위험 상황을 제거하고 비상정지 스위치와 보호정지 장치의 상태를 확인해야 합니다.

3) 어떠한 경우에도 사용자는 로봇 동작 전에 로봇의 설치 상태를 올바르게 파악하여 설정 완료를 하여야 합니다.

■ Initialize Mode [Set-up]

로봇 동작 전에 주변 장치 설정이나 로봇의 상태를 설정할 수 있습니다.

작업 공간, TCP 오프셋, 가반하중 등 로봇 동작에 있어 전반적인 설정을 진행할 수 있습니다.

이때 모터에는 전원이 공급되어 있지 않으며, 활성화 작업을 통해서 모터에 전원을 공급할 수 있습니다.

■ Auto Mode [Play]

로봇이 사용자의 간섭 없이 미리 정의된 작업만 수행하는 상태입니다. 이때 모터에는 전원이 공급되어 있습니다.

Simulation 기능을 통해 프로그래밍 된 로봇의 동작을 시뮬레이션을 통해 검증할 수 있으며, Real 상태로 변환하여 검증된 프로그램으로 로봇을 구동 할 수 있습니다.

이때 로봇은 사전에 설정된 안전 기능이 수행되고 있으며, 사용자는 Play 창을 통해 로봇, 주변 기기의 상태를 모니터링 할 수 있습니다.

■ Manual Mode [Make]

로봇이 사용자의 직접적인 행위를 통해 동작하는 상태입니다. 이때 모터에는 전원이 공급되어 있습니다.

직접 교시, 프로그램 생성 및 수정, 주변 기기 수동 동작 등을 수행할 수 있으며, 안전 속도 슬라이드 바를 통해 사용자가 PC 태블릿을 조작하는 순간에만 로봇이 동작할 수 있습니다.

안전 슬라이드에서 손을 떼면 로봇의 동작은 즉시 정지하도록 되어 있습니다.

주의

주의:

1) 직접 교시의 경우에 안전 슬라이드 기능을 설정한 상태로 사용해야 합니다.
2) 초기 출고 시 안전 슬라이드 기능은 비활성화 되어 있습니다.
3) 추가적으로 3 위치 활성화 스위치를 사용 시에 ISO 10218-1 의 5.8.3 을 준수하여 사용해야 합니다.

3.7 사용 환경

장기간 로봇을 안전한 상태로 유지하기 위해서는 아래와 같은 환경에서 사용해야 합니다.

최대 허용 동작 온도	50˚C
최대 허용 보관 온도	60˚C
최소 허용 작동 온도	0˚C
최소 허용 보관 온도	-5˚C
최대 허용 습도	80%
최저 허용 습도	20%

3.8 안전 기능의 유지관리

장기간 로봇을 안전한 상태로 유지하기 위해서는 지속적인 안전기능의 점검이 필요합니다.

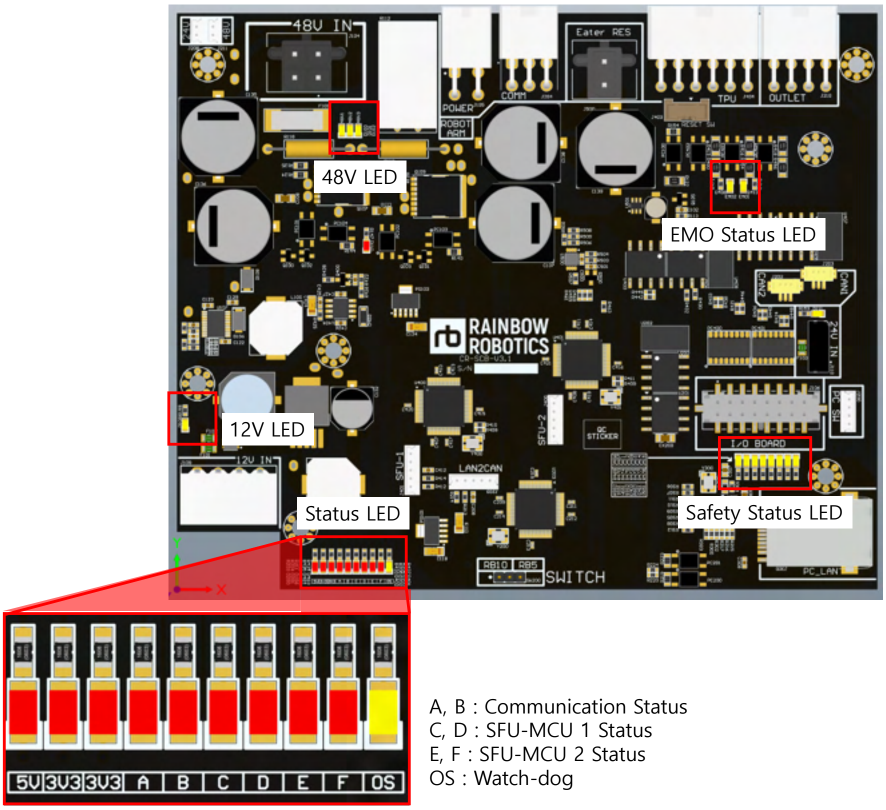
안전 점검 관리자는 다음과 같은 항목에 대한 주기적인 점검이 필요합니다. 점검 중 자체적으로 해결할 수 없는 문제를 발견했을 때는 제조사에 문의하십시오.

점검 항목		점검 사항	주기
펜던트	안전 기능	펜던트에 장착된 비상정지 스위치의 정상 작동 유무를 확인하십시오.	1 개월
	케이블	펜던트와 컨트롤 박스 간 연결 케이블의 상태를 확인하십시오.	1 개월
컨트롤 박스	인터페이스	안전 장치를 연결하는 EMO 포트의 정상 작동 유무를 확인하십시오.	1 개월
		안전 장치를 연결하는 PRS 포트의 정상 작동 유무를 확인하십시오.	1 개월
		안전 장치를 연결하는 HSS 포트의 정상 작동 유무를 확인하십시오.	1 개월
		안전 장치를 연결하는 SSS 포트의 정상 작동 유무를 확인하십시오.	1 개월
	전원	안전 장치를 연결하는 24V 전압의 정상 출력을 확인하십시오.	1 개월
		24V 퓨즈가 정상적으로 삽입되어 있는지 확인하십시오.	1 개월
	케이블	안전 장치와 컨트롤 박스 간 연결 케이블의 상태를 확인하십시오.	1 개월

■ 안전 기능 보드 사양

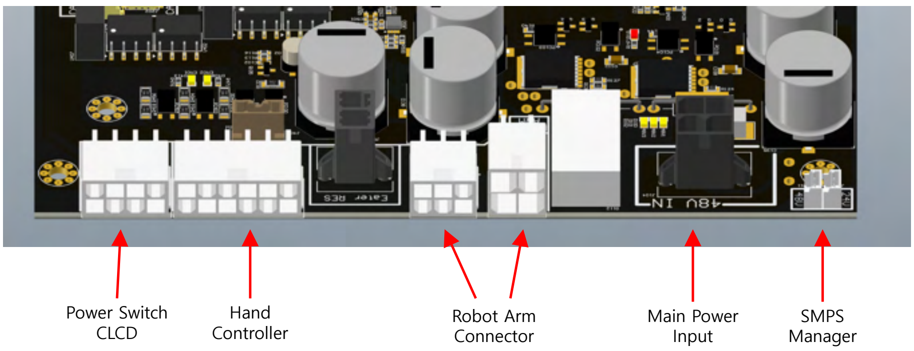
컨트롤 박스 내부에는 협동로봇 RB Series를 구동하기 위한 안전 기능 보드가 내장되어 있습니다.

보드의 동작상태를 알려주는 LED의 정보는 아래와 같습니다.



A, B : Communication Status
C, D : SFU-MCU 1 Status
E, F : SFU-MCU 2 Status
OS : Watch-dog

보드와 연결되는 커넥터 정보는 다음과 같습니다.



3.9 적용 표준

표 준	표 제
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 1: General requirements
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 2: Requirements for electrical/electronic/programmable electronic safetyrelated systems
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 3: Software requirements
IEC 61508-4:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 4: Definitions and abbreviations
IEC 61508-5:2010	Functional safety of electrical/electronic/programmable electronic safety-related system – Part 5: Examples of methods for the determination of safety integrity levels
IEC 61508-6:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 6: Guidelines on the application of IEC 61508-2 and IEC 61508-3
IEC 61508-7:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 7: Overview of techniques and measures
IEC 60204-1:2016	Safety of machinery – Electrical equipment of machines – Part 1: General requirements
IEC 61000-6-1: 2016	Electromagnetic compatibility (EMC) – Part 6-1: Generic standards – Immunity standard for residential, commercial and light-industrial environments

IEC 61000-6-2: 2016	Electromagnetic compatibility (EMC) – Part 6-2: Generic standards – Immunity standard for industrial environments
IEC 61000-6-7: 2014	Electromagnetic compatibility (EMC) – Part 6-7: Generic standards – Immunity requirements for equipment intended to perform functions in a safety-related system (functional safety) in industrial locations
IEC 61326-3-1: 2017	Electrical equipment for measurement, control and laboratory use – EMC requirements – Part 3-1: Immunity requirements for safety-related systems and for equipment intended to perform safety-related functions (functional safety) – General industrial applications
IEC 61800-5-1: 2007	Adjustable speed electrical power drive systems – Part 5-1: Safety requirements –Electrical, thermal and energy
IEC 61800-5-2: 2016	Adjustable speed electrical power drive systems – Part 5-2: Safety requirements – Functional
IEC 62061:2005	Safety of machinery – Functional safety of safety-related electrical, electronic and programmable electronic control systems
ISO/TS 15066: 2016	Robots and robotic devices — Collaborative robots
ISO 10218-1: 2011	Robots and robotic devices — Safety requirements for industrial robots — Part 1: Robots
ISO 10218-2: 2011	Robots and robotic devices — Safety requirements for industrial robots — Part 2: Robot systems and integration
ISO 12100:2010	Safety of machinery — General principles for design — Risk assessment and risk reduction
ISO 13849-1: 2015	Safety of machinery — Safety-related parts of control systems — Part 1: General principles for design
ISO 13849-2: 2012	Safety of machinery — Safety-related parts of control systems — Part 2: Validation

제 4 장 설치하기

4.1 설치할 때 유의사항

로봇 설치자는 ISO 12100 및 ISO 10218-2 의 지침에 따라 로봇을 설치하고 운영해야 하며, ISO/TS 15066 와 같은 국제 표준 및 국내 법규의 관련 요건을 따라야 합니다. 국제 표준 및 국내 법규의 관련 요건을 따르지 않거나, 앞 장에서 언급한 위험성 평가를 검토하지 않은 경우에 발생하는 사고는 제조사가 책임지지 않습니다.

당사의 로봇은 "산업안전 보건기준에 관한 규칙" 및 "안전검사 고시"의 검사 기준을 고려하여 설치하여야 합니다. 자세한 내용은 부록 M 의 내용을 참조하시기 바랍니다.

4.2 설치 장소

다음의 조건을 만족하는 장소에 시스템 설치를 권장합니다.

- 내진 설계가 되어있는 건물
- 누수가 전혀 없는 곳
- 인화성 및 폭발 물질이 없는 곳
- 온도와 습도가 일정하게 유지될 수 있는 곳
- 먼지 유입이 많지 않은 곳

주의

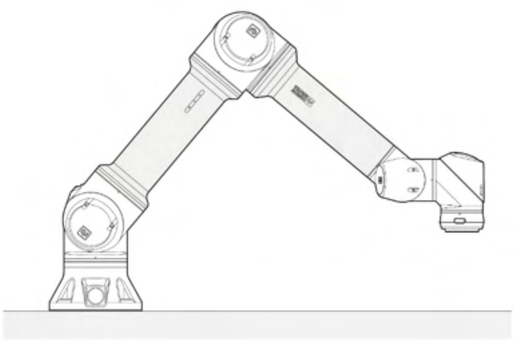
주의:

1) 권장하는 장소에 시스템을 설치하지 않을 경우 로봇의 성능 및 수명이 감소될 수 있습니다.

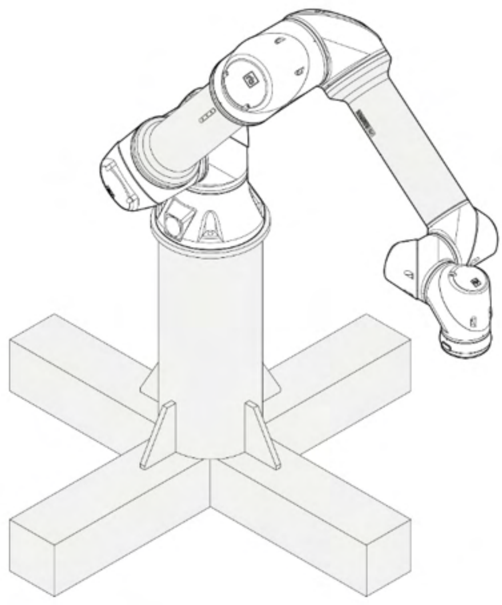
4.3 설치 유형

로봇 팔은 바닥과 벽면 등 원하는 곳에 설치할 수 있으며, 고정 포스트에도 설치가 가능합니다. 고정 포스트는 제품에 포함되어 있지 않습니다.

이외에도 벽면, 천장, 또는 임의의 각도로 설치 가능합니다. 평면 설치 이외의 설치는 시스템 설정을 통해 설치 각도를 입력해주어야 합니다.



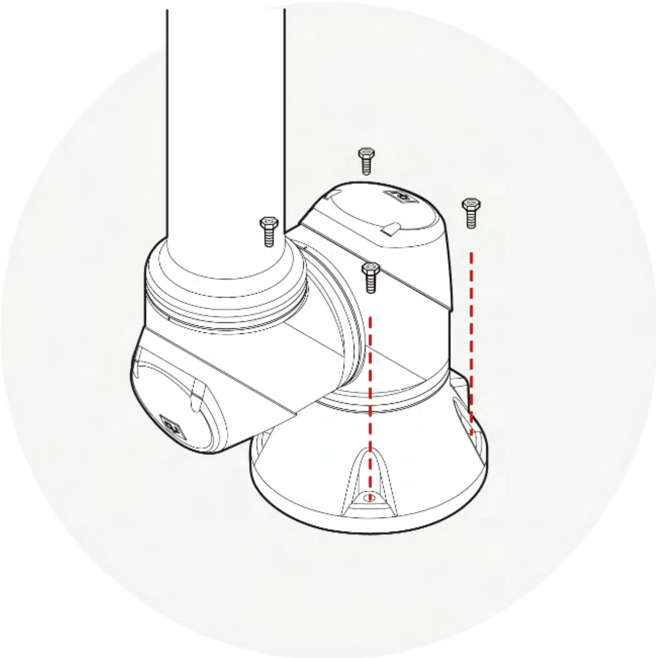
[바닥 설치의 예]



[고정 포스트 설치의 예]

4.4 로봇 고정하기

로봇 팔은 M8 30 mm 볼트 4 개를 이용하여 설치하는 것을 권장합니다.



경고

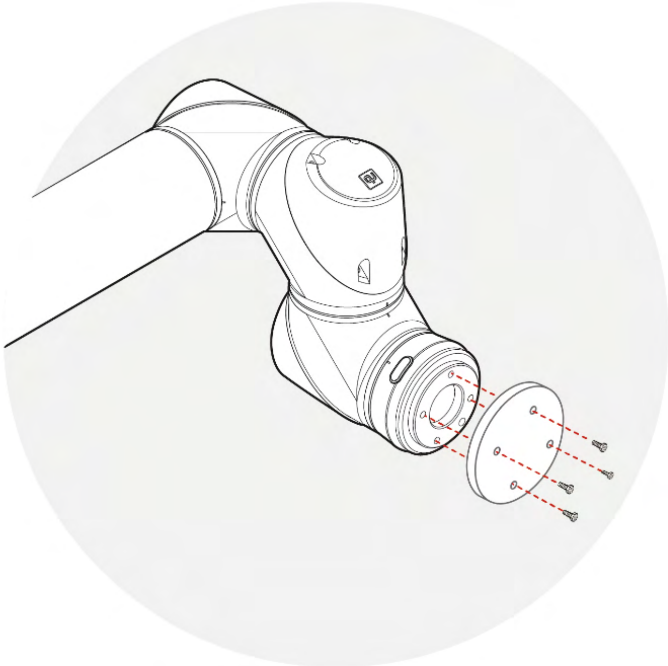
경고:

1) 로봇을 고정할 때에는 볼트가 풀리지 않도록 단단히 고정하십시오.
2) 로봇의 무게와 로봇이 작동할 때 발생하는 하중을 견딜 수 있는 견고한 면에 설치하십시오.
3) 로봇 팔의 설치면이 베이스면과 완전히 밀착되었는지 확인하십시오.
4) 로봇에 조립되어 있는 볼트는 절대 임의로 분해하지 마십시오. 로봇 팔을 작동하기 전에 모든 볼트가 잘 고정되어 있는지 확인하십시오.
5) 볼트가 분해된 상태에서 사용하거나 볼트 조립부를 이용하여 브라켓 등을 설치할 경우 제품이 파손되거나 안전에 심각한 영향을 미칠 수 있습니다.

4.5 로봇과 툴 연결하기

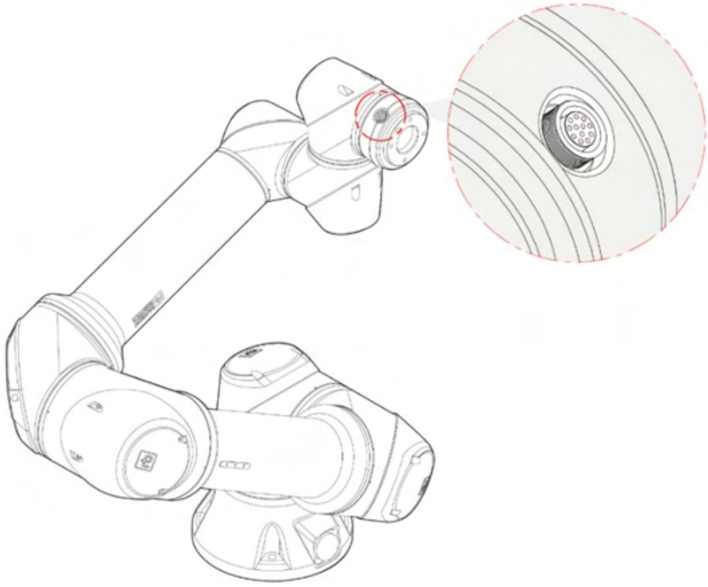
M6 볼트 4 개를 이용하여 툴 플랜지에 툴을 고정합니다.

- 툴과 고정용 M6 볼트는 제품에 포함되지 않습니다.
- 툴을 결합하는 방법은 툴마다 다를 수 있습니다. 자세한 방법은 툴 제조사에서 제공하는 사용설명서를 참조하십시오.



- 툴을 고정한 후 툴 I/O 또는 컨트롤 박스에 있는 I/O 포트에 필요한 케이블을 연결하십시오.
- 툴 커넥터는 삼우 커넥터사의 SW-10W-12(P)를 사용합니다.

■ 툴 I/O 는 12 핀 커넥터가 있습니다.



■ 툴 플랜지에 위치하고 있는 포트 사양은 Non-E Version 과 E Version 으로 나뉘고 이는 다음과 같습니다.

포트	레이아웃	핀번호	시그널 (Non-E Version)	시그널 (E Version)	색상 (Thickness)
Tool I/O		1	Digital Output A	Digital Output A	**갈색 (AWG22)**
		2	Digital Output B	Digital Output B	**파랑색 (AWG22)**
		3	0/12/24 VCC	0/12/24 VCC	**빨강색 (AWG22)**
		4	Ground	Ground	**검정색 (AWG22)**
		5	Digital Input A	Digital Input A	흰색 (AWG26)
		6	Digital Input B	Digital Input B	파랑색 (AWG26)
		7	Analog Input A	Digital Input C	흰색 (AWG26)
		8	Analog Input B	Digital Input D	노란색 (AWG26)
		9	RS485+	RS485+	회색 (AWG26)
		10	RS485-	RS485-	자주색 (AWG26)
		11	Common Ground	Digital Input E	검정색 (AWG26)
		12	Common Ground	Digital Input F	녹색 (AWG26)

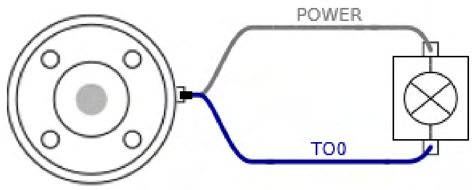
■ 내부 전원 공급은 GUI 의 I/O 탭에서 0V, 12V, 24V 로 설정할 수 있습니다.

-	최소	형식	최대	단위
24V 모드의 공급전압	-	24	-	V
12V 모드의 공급전압	-	12	-	V
두 모드의 공급전류	-	-	2000	mA

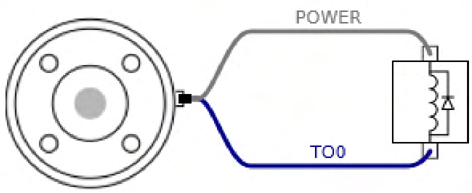
■ 툴 디지털 출력은 NPN 방식으로 적용되어 있습니다. 디지털 출력을 활성화하면, 해당 포트의 연결은 GND(ground)로 이어집니다. 비활성화 시, 해당 출력 포트는 개방됩니다(오픈-컬렉터/오픈-드레인). 전기 사양은 다음과 같습니다.

-	최소	형식	최대	단위
개방 시 전압	0	-	24	V
GND 를 통한 전류	0	-	2000	mA

■ 툴 디지털 출력을 사용하기 위해서는 아래 그림과 같이 12V 또는 24V 전원 공급을 사용하여 부하를 켜는 것을 보여줍니다. Tool Out 블록에서 출력 전압을 정의할 수 있습니다.



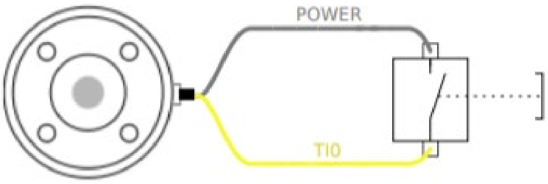
※ 아래 그림에 표시된 대로 유도 부하에 대하여 보호 다이오드를 사용하시길 적극 권장합니다.



■ 툴 디지털 입력은 PNP 및 풀 다운저항이 적용되어 있습니다. 따라서 입력 단자에 연결이 없을 경우(floating) 해당 입력 포트는 로우(Low, 0)로 읽힙니다. 전기 사양은 다음과 같습니다.

-	최소	형식	최대	단위
입력전압	0	-	24	V
논리적 저 전압	-	-	9	V
논리적 고 전압	10	-	-	V

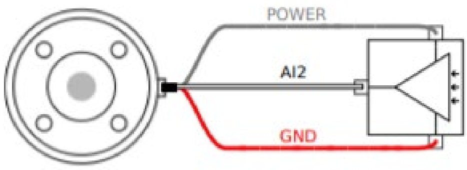
■ 툴 디지털 입력을 사용하기 위하여 아래의 그림은 간단한 버튼을 연결하는 방법을 보여줍니다.



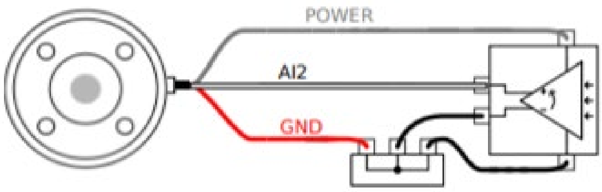
■ 툴 아날로그 입력은 비차등(non-differential) 방식으로 전압을 측정하며, 측정 범주는 아래와 같습니다.

-	최소	형식	최대	단위
전압모드의 입력전압	0	-	10	V
분해능	-	12	-	bit

■ 아래의 그림은 비차등(non-differential) 전압 출력 특성을 가진 아날로그 센서를 툴 플랜지에 연결하는 방법을 보여줍니다. (비차등 방식)



■ 아래의 그림은 차등(differential) 전압 출력 특성을 가진 아날로그 센서를 툴 플랜지에 연결하는 방법을 보여줍니다. 센서의 음출력부(negative output part)를 GND(ground)에 연결하면 비차등 센서와 동일하게 작동됩니다.



주의

주의:

1) 자세한 기술 사양과 결선 예시는 부록 D 에 표기되어 있습니다.
2) 툴 플랜지의 단면도에 관한 사항은 부록 C 를 참조하시기 바랍니다.

■ 툴 플랜지에서는 RS485 시리얼 통신을 지원하며, 아래와 같은 시리얼 통신 규격을 지원합니다.

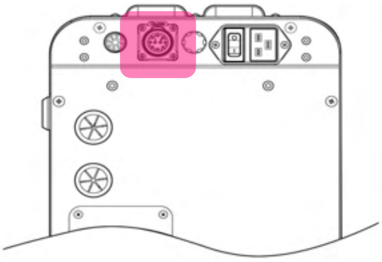
전송속도	9600, 19200, 38400, 57600, 115200, 1M, 2M.
정지비트	1,2
패리티	없음, 홀수, 짝수

4.6 케이블 커넥션

스탠드 타입 컨트롤 박스의 케이블 커넥션은 다음과 같습니다.

■ **로봇 컨트롤 박스와 로봇 팔 케이블 연결하기**

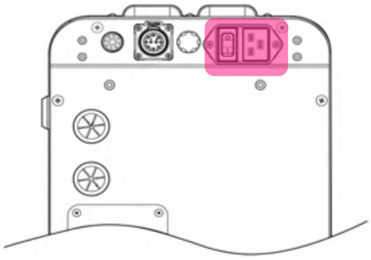
로봇 팔에는 암 커넥터를, 로봇 컨트롤 박스와 수 커넥터를 연결합니다. 커넥터 연결 시 Pin 휨에 주의합니다.



[로봇 팔 케이블 연결 부분]

■ **로봇 컨트롤 박스에 전원 연결하기**

전원 단자에 해당하는 전원 케이블을 연결합니다.



[AC 전원단자 연결 부분]

전원부의 전기사양은 다음과 같습니다.

입력 전원	100 ~ 240 VAC
입력 주파수	50 ~ 60 Hz

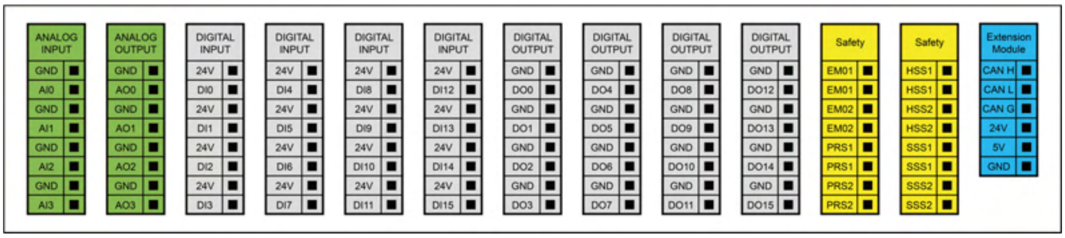
주의

주의:

1) 로봇이 켜져 있는 상태에서는 로봇 커넥터, 전원 커넥터, 태블릿 PC 커넥터 등을 뽑지 마십시오.
2) AC 전원은 주변부와 접점 (Ground)이 잡혀 있어야 합니다.

4.7 로봇 컨트롤 박스 I/O 개요

외부의 다른 장비를 로봇 컨트롤 박스와 연결하려면 로봇 컨트롤 박스 안에 있는 I/O 와 해당 장비를 연결합니다. 로봇 컨트롤 박스의 I/O 는 매우 유연하므로 릴레이, PLC, 비상 정지 버튼 등 다양한 장비와 연결하여 사용할 수 있습니다. 로봇 컨트롤 박스 내부의 전기 인터페이스 레이아웃은 다음과 같습니다.



컨트롤 박스에서 제공하는 전원과 디지털 I/O 의 사양은 다음과 같습니다.

모든 디지털 I/O 는 IEC 61131-2 규격에 호환됩니다.

※IEC 61131-2: 프로그래머블 컨트롤러를 위한 IEC 표준

터미널	매개변수	최소	형식	최대	단위
디지털 출력					
[Dox]	전류	0	-	1	A
[Dox]	전압 강하	0	-	0.5	V
[Dox]	누설 전류	0	-	0.1	mA
[Dox]	장치 형태	-	PNP		Type
[Dox]	IEC 61131-2	-	1A	-	Type
디지털 입력					
[Dix]	전압	-3	-	30	V
[Dix]	OFF 범위	-3	-	5	V
[Dix]	ON 범위	11	-	30	V
[Dix]	전류 (11-30V)	2	-	15	mA
[Dix]	장치 형태		PNP+		Type
[Dix]	IEC 61131-2	-	1	-	Type

주의

주의:

I/O 배선 체결 시, 컨트롤 박스 전원을 끄고 I/O 커넥터에 미리 작업하도록 합니다. 사용자의 부주의(24V 전원 쇼트 및 잘못된 배선 등)로 인한 제품 손상은 제품의 무상 보증을 받으실 수 없습니다.

4.8 안전 입력 구성

사용시 안전을 위해, 안전과 관련된 모든 I/O 는 중복으로 구성해야 합니다. 안전 장치 및 장비는 2 장 안전 및 3 장 설치의 지시 사항을 준수하여 설치해야 합니다. 안전 입력에는 안전 보호 정지가 있습니다. 비상 정지 입력은 로봇의 비상 정지를 위한 것이며, 안전 보호 정지 입력은 로봇의 모든 안전 등급 보호를 위한 것입니다.

위험

위험:

1) 안전 PLC 가 아닌 PLC 에는 절대로 안전 신호를 연결하지 않습니다. 해당 경고를 따르지 않으면, 안전 정지 기능이 정상적으로 작동되지 않을 수 있으므로 심각한 부상 또는 사망에 이를 수 있습니다. 안전 신호와 일반 I/O 신호는 분리되어야 합니다.

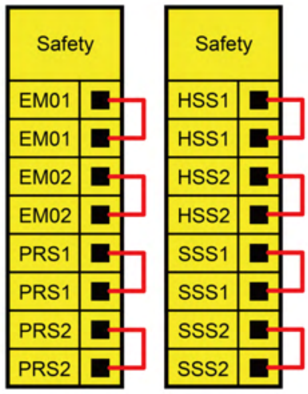
경고

경고:

1) 모든 안전 등급의 입출력은 이중화 되어 있습니다. 해당 채널을 분리하여 신호 장애로 안전 기능이 작동되지 않도록 하여야 합니다. 안전 기능은 로봇을 설치하기 전 반드시 확인을 해야 합니다. 또한 해당 기능은 정기적으로 이상유무를 점검해야 합니다.

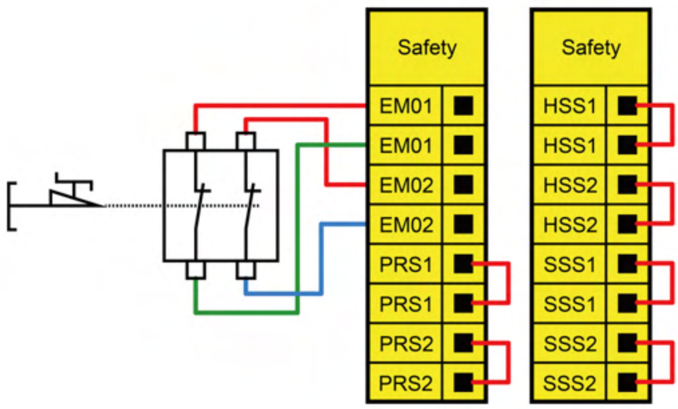
■ 초기 안전 구성

로봇은 초기 안전 구성이 기본 값으로 설정이 되어 배송이 되므로 추가 설정 없이 구성할 수 있습니다. 초기 안전 구성은 다음과 같습니다.



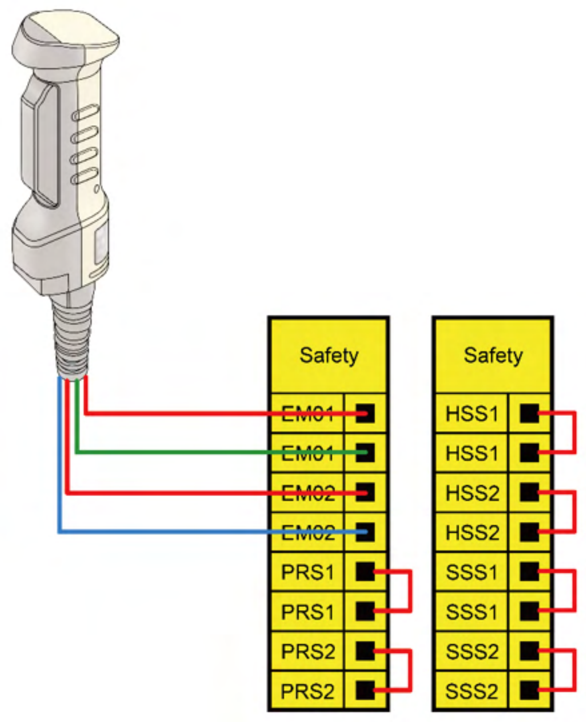
■ 안전 보호 정지 및 자동 재시작

기본적인 안전 보호 정지 장치의 예로 도어가 열릴 경우 로봇을 정지하는 도어 스위치가 있습니다. 해당 기능을 구성하는 방법은 다음과 같습니다.



■ Enabling Device 입력(Option)

사용자는 3 위치 활성화 스위치를 사용하여 활성화 장치 입력 인터페이스를 연결할 수 있습니다. 3 위치 활성화 스위치가 작동 위치(중간 위치)에 있을 때 로봇이 움직이기 시작하며, 사용자가 3 위치 활성화 스위치를 놓거나 누를 때 3 위치 활성화 스위치는 비작동 위치에 있고 로봇 팔은 움직이지 않습니다. 제조사는 Enabling Device 를 제공하지 않으며, 사용자가 Enabling Device 를 사용하고자 하는 경우에 옵션으로 사용할 수 있습니다. 해당 기능을 구성하는 방법은 다음과 같습니다.

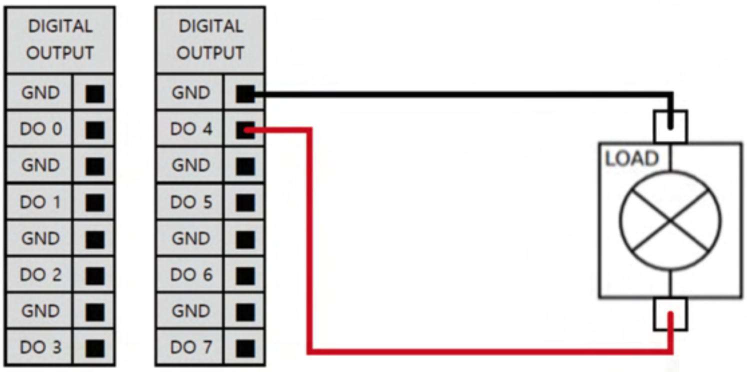


4.9 범용 디지털 I/O 구성

모든 Digital I/O 는 범용 디지털 I/O 로 사용할 수 있습니다. 외부 다른 장비를 로봇과 함께 사용하기 원한다면 로봇 컨트롤 박스에 있는 I/O 와 해당 장비를 연결해야 합니다. 범용 디지털 I/O 는 릴레이와 같은 장비 또는 PLC 시스템을 구성하는데 사용할 수 있습니다. 이와 같은 구성에서는 프로그램이 구동 중이 아니라면 출력을 항상 LOW 로 유지합니다. 다음 하위 항목들은 예를 보여주고 있습니다.

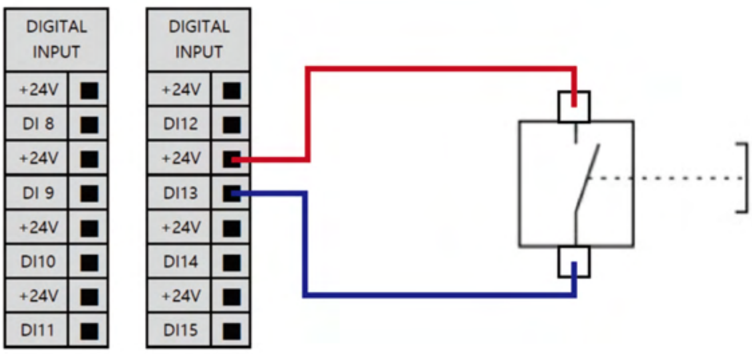
■ **디지털 출력으로 부하 제어**

디지털 출력으로 부하를 제어할 수 있는 방법입니다. 구성 방법은 다음과 같습니다.



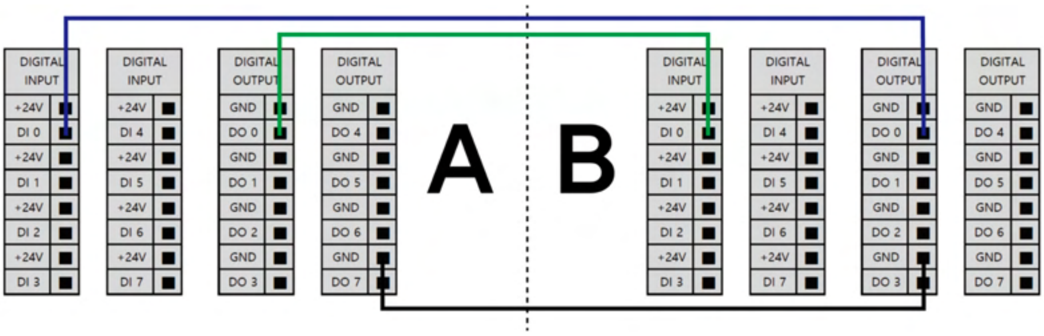
■ **버튼을 이용한 디지털 입력 제어**

디지털 입력에 간단한 버튼을 연결하는 방법입니다. 구성 방법은 다음과 같습니다.



■ 다른 장비 또는 PLC 와 통신

공통 GND 를 사용하고, 장비가 PNP 를 지원하면, 디지털 I/O 를 다른 장비와 통신하도록 구성할 수 있습니다. 구성 방법은 다음과 같습니다.



주의

주의: 자세한 기술 사양과 결선 예시는 부록 D 에 표기되어 있습니다.

4.10 범용 아날로그 I/O 구성

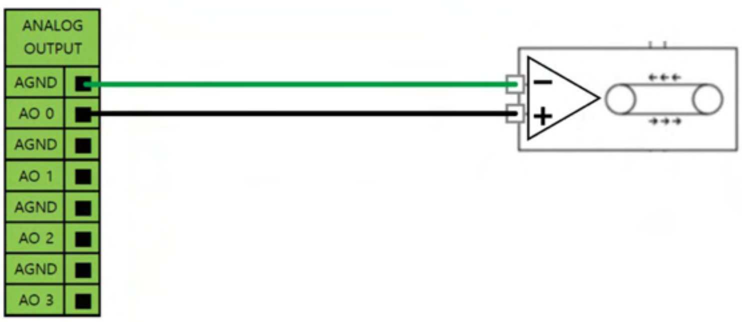
높은 신뢰성을 위하여 다음과 같은 방법을 권장합니다.

- I/O 에 가장 가까운 아날로그 GND 를 사용합니다.
- 장비 및 로봇 컨트롤 박스는 동일한 GND 를 사용합니다. 아날로그 I/O 는 로봇 컨트롤 박스에서 절연되지 않습니다.
- 쉴드 또는 트위스트 페어 처리된 케이블을 사용합니다. Power(J12) 터미널의 GND 쉴드에 연결합니다.

터미널	매개변수	최소	형식	최대	단위
전압모드 입력					
AIx - AG	전압	0	-	10	V
AIx - AG	분해능	-	16	-	비트
전압모드 출력					
AOx – AG	전압	0	-	10	V
AOx – AG	분해능	-	16	-	비트

■ **아날로그 출력**

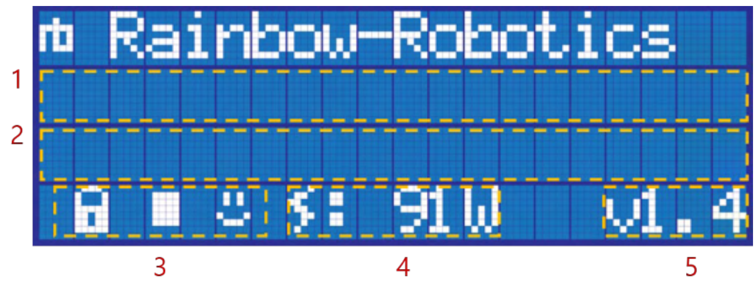
아날로그 출력을 컨베이어 벨트의 속도 컨트롤 입력으로 사용할 수 있습니다. 구성 방법은 다음과 같습니다.



■ 아날로그 입력

아날로그 센서의 출력 값을 아날로그 입력으로 사용할 수 있습니다. 구성 방법은 다음과 같습니다.

■ LCD 상태 표시창 설명



1. 알림 박스(1): 시스템 상태에 관한 알림 박스입니다.

   : Please Wait (컨트롤 박스 내부 PC 가 준비가 되지 않았음을 나타냅니다.)

   : Normal Operation (컨트롤 박스 내부 PC 가 준비가 끝났음을 나타냅니다.)

2. 알림 박스(2): 로봇 동작 및 상태에 관한 알림 박스입니다.
3. 동작 아이콘: 브레이크 잠김() 또는 해제() 상태, 재생() 또는 정지() 상태, 충돌() 또는 안전() 상태를 나타냅니다.
4. 전력 사용량: 와트(W) 단위의 총 전력 사용량을 나타냅니다.
5. 시스템 버전 정보: 시스템 버전 정보입니다.

제 5 장 시작하기

5.1 로봇 컨트롤 박스 켜기/끄기

로봇 컨트롤 박스의 켜기/끄기 방법은 다음과 같습니다.

- **스탠드 타입 컨트롤 박스 켜기/끄기**

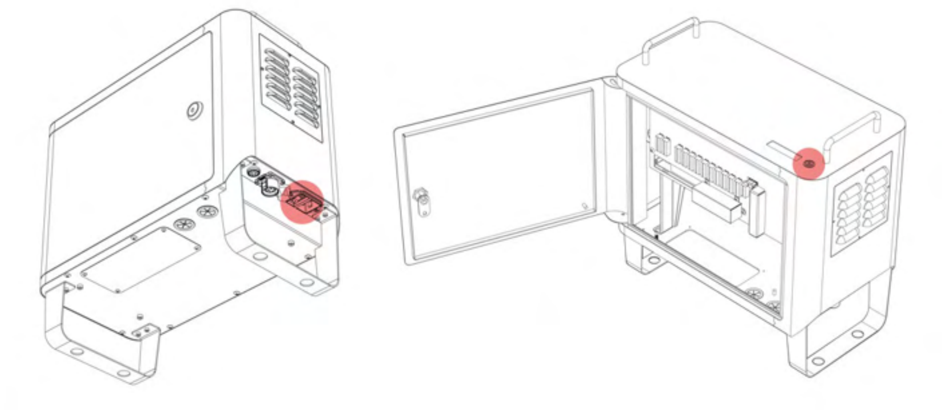
컨트롤 박스 하단의 AC 전원 스위치를 누르면 AC 전원이 인가됩니다.

컨트롤 박스 상단의 메인 전원 스위치를 누르면 주 전원이 켜집니다.

로봇 컨트롤 박스의 LCD 창에 "Please Wait"가 표시됩니다.

이는 컨트롤 박스 초기 부팅 상태임을 나타냅니다.

컨트롤 박스가 사용가능 상태로 변경되면 LCD 메세지는 "Normal Operation" 로 변경됩니다.



컨트롤 박스 상단의 메인 스위치를 길게 누르고 있으면 전원이 꺼집니다.

주의

주의: 당사의 컨트롤 박스는 AC 100~240V 단상 (50~60Hz)을 사용합니다.

5.2 태블릿 PC 켜기

제조사에서 제공하는 컨트롤 박스용 태블릿 PC 옵션을 선택한 경우, 아래와 같은 태블릿 PC 와 커버가 제공됩니다. 태블릿 PC 에 전원을 켜시려면 태블릿 PC 좌측 상단의 전원 버튼을 누르십시오.

- **스탠드 타입 컨트롤 박스용**

주의

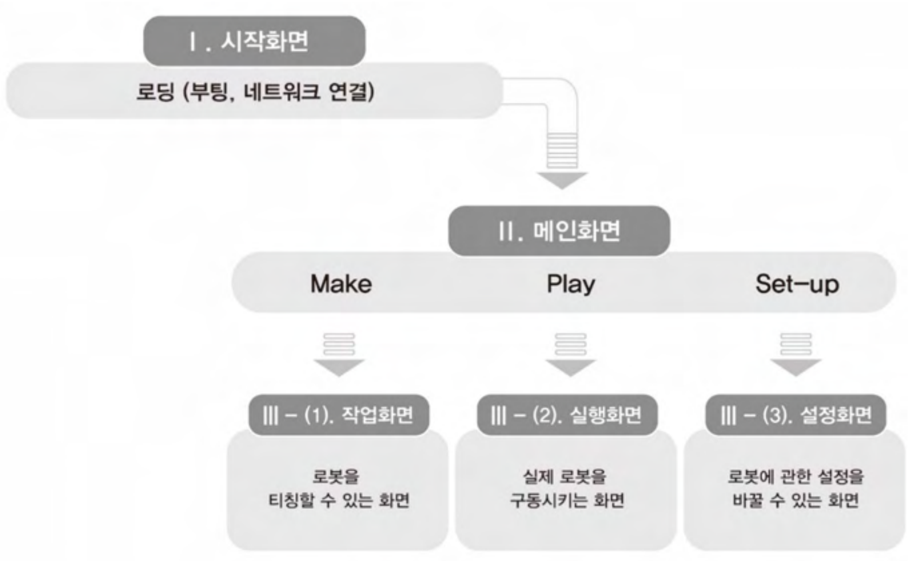
주의:

1) 태블릿 PC 가 컨트롤 박스와 연결되어 있는지 확인하시고 제조사에서 제공한 어플리케이션을 실행하시기 바랍니다. 시스템이 부팅하는 중에는 화면에 불필요한 조작을 하지 마십시오. 시스템에 문제가 발생할 수 있습니다.

제 6 장 소프트웨어 개요

6.1 UI 프로그램 기본구성

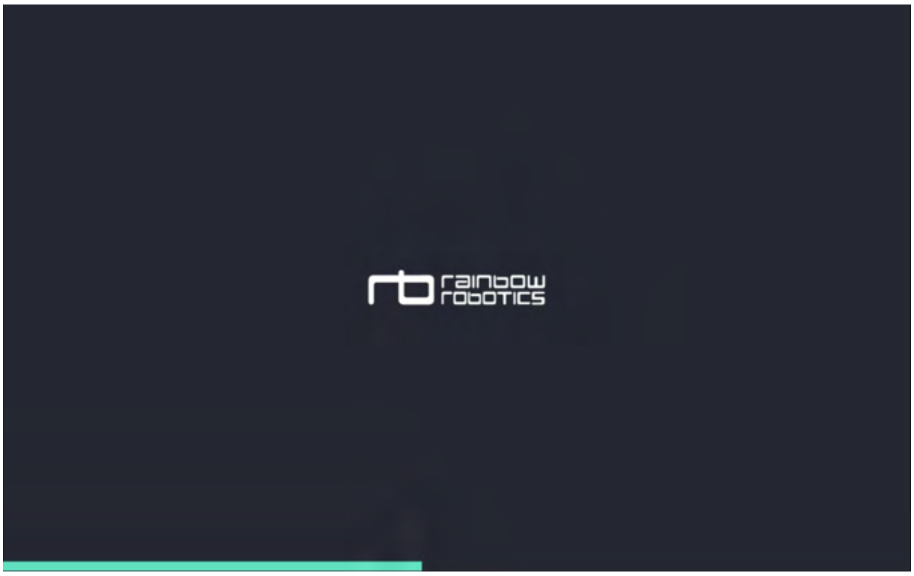
UI(User Interface) 프로그램은 크게 다음과 같이 3 가지 화면으로 구분되며, 각 영역을 통해 사용자는 필요한 단계로 진입할 수 있습니다.



6.2 시작 화면

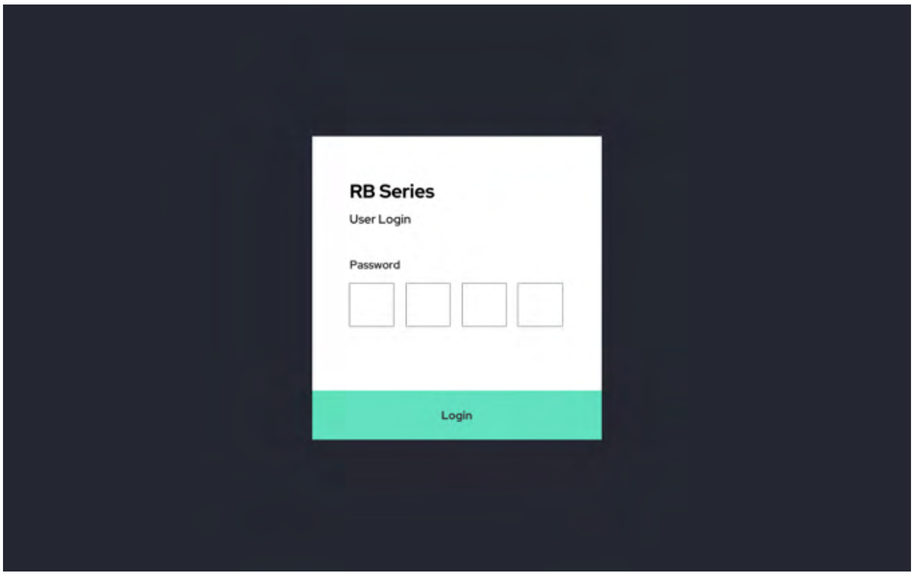
■ 인트로

시작 화면에서는 인트로 화면을 통해 로딩 과정이 이루어집니다.



■ 로그인(<u>공장 출하 시 Password: 0000 으로 설정되어 있음</u>)

시작 시 로그인 확인을 합니다. 비밀번호 설정과 자동 로그인 설정은 **"Setup-System-Password"**에서 진행합니다.

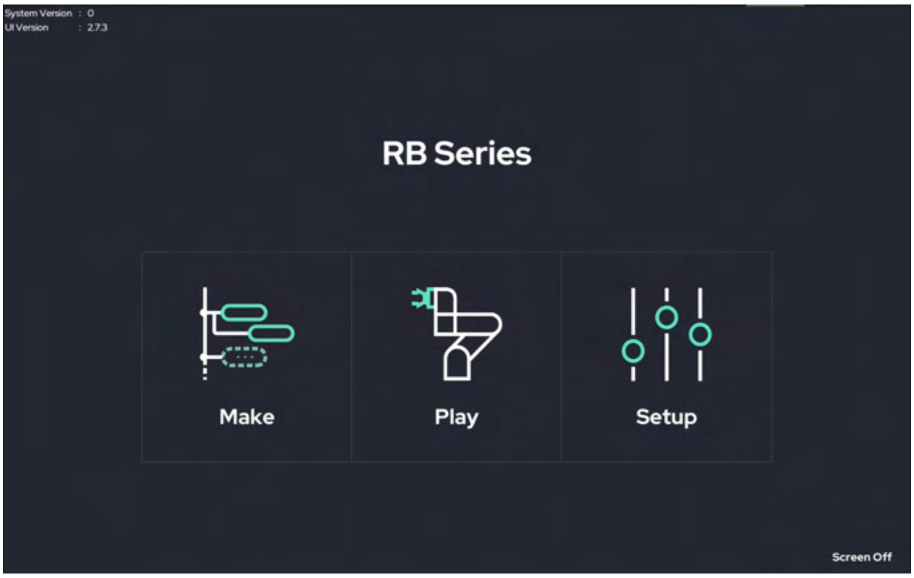


6.3 메인 화면

UI 는 아래와 같이 3 가지 메인 메뉴로 구성됩니다.

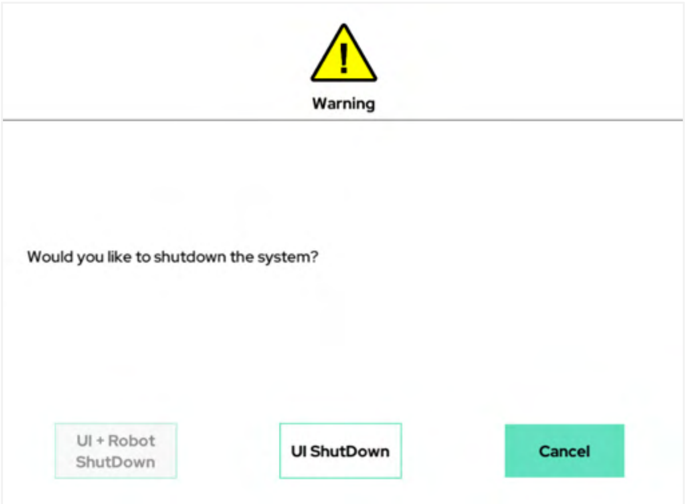
- Make: 프로그램(작업 시퀀스)을 실제로 만드는 영역입니다.
- Play: 만들어진 프로그램을 단순 실행하는 영역입니다.
- Setup: 각종 파라미터를 설정하는 영역입니다.

메인 화면에서는 3 개의 메뉴를 통해 로봇의 움직임을 만들고(Make), 실제로 움직이거나(Play), 작업 환경을 설정(set-up)하는 단계로 진입할 수 있습니다.



■ **전원 종료**

우측하단의 전원 버튼을 터치하면 전원 종료창이 나타나며 UI Shutdown 을 누를 경우에 프로그램이 종료됩니다. 만약 로봇이 활성화되어 있고, 컨트롤 박스가 UI 태블릿 PC 와 연결이 되어 있는 경우, 로봇의 전원도 함께 내려갑니다.



6.4 작업 화면(MAKE)

■ Make

로봇을 티칭(teaching)할 수 있는 화면입니다. 화면의 상단에 있는 아이콘들을 이용하여 로봇 팔을 티칭할 수 있습니다. 화면 오른쪽에 나열된 아이콘들을 사용하여 로봇 팔을 움직일 수 있고, 왼쪽에 나열된 아이콘들을 사용하여 티칭한 내용을 편집할 수 있습니다.

- **좌측**: 프로그램 구조 속 내용을 복사/붙여넣기/저장/삭제/주석화 등 편집하는 기능이 있습니다.
- **우측**: 기본적인 로봇 조깅(Jog/Jogging) 및 기타 설정이 있습니다.
- **중간**: 프로그램을 구성할 수 있는 기능들이 배치되어 있습니다.
- **하단**: 프로그램 저장/불러오기, 재생, 속도 조절 바 등 인터페이스 요소가 있습니다.

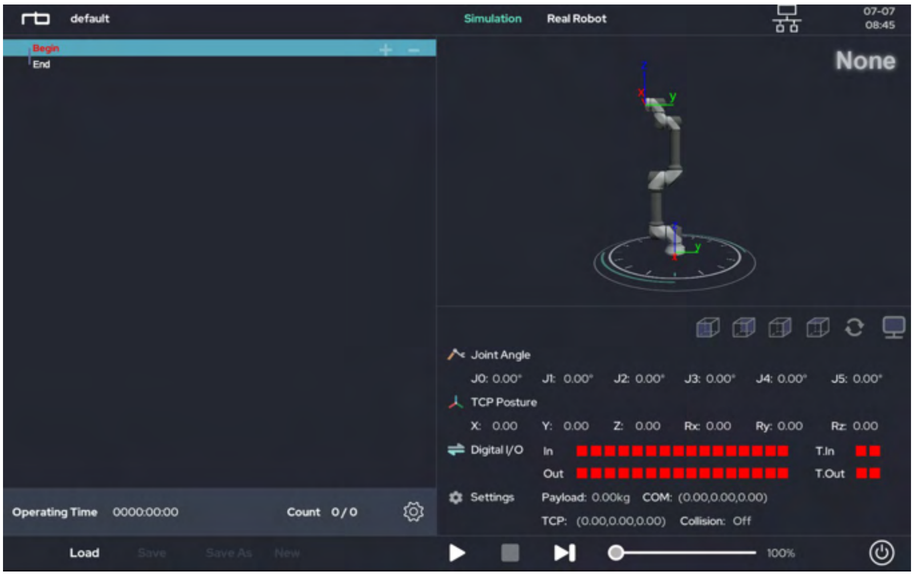
※ 각각의 아이콘들과 화면 구성에 관한 설명은 7 장에 더욱 자세하게 설명되어 있으니 참고하시기 바랍니다.

※ Make 메뉴(Manual mode)에서는 로봇 동작에 관련되는 버튼을 누르고 있는 상태에서만 로봇이 동작하게 되며, 버튼을 누르지 않은 상태에서는 로봇이 동작하지 않습니다. 이러한 기능은 Setup-Interface 에서 설정 값을 변경할 수 있습니다.

6.5 실행 화면(PLAY)

- Play

로봇 팔을 실제로 구동 시키는 화면입니다. 이 화면에서는 티칭한 파일(프로그램)을 불러들여 실행할 수 있습니다. Play 반복 횟수는 Setup-Interface 에서 설정가능 합니다. 좌측 하단에는 초기 재생 이후 누적 시간이 표기됩니다.

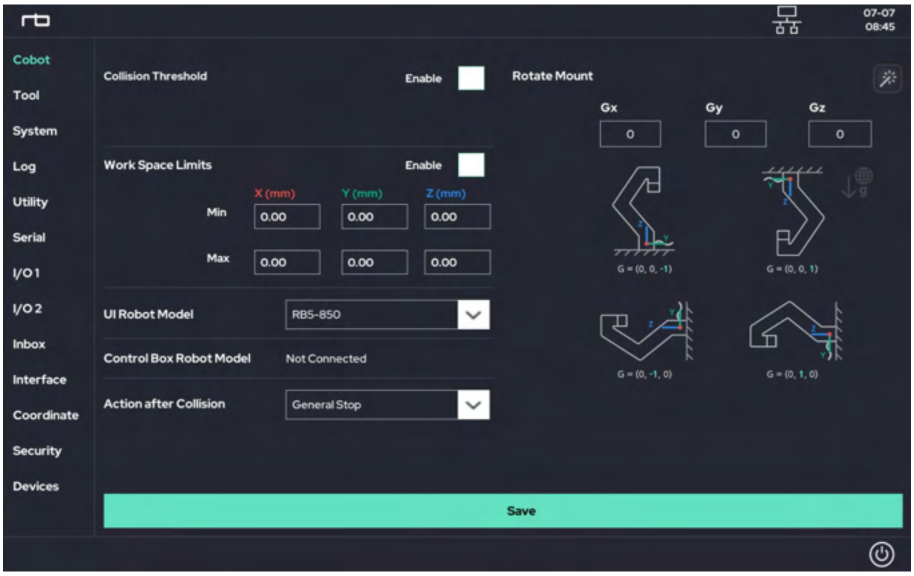


※ 더욱 자세한 내용 설명은 8 장에 설명된 내용을 참고하시기 바랍니다.

6.6 설정 화면(SETUP)

- Setup

로봇 팔을 사용하기 위해 다양한 설정을 할 수 있는 화면입니다. 충돌 감도, 로봇 설치 각도, 작업 영역, 툴 세팅 파라미터, 시스템 로그 뷰, I/O 기능 설정, 좌표계 설정 등 다양한 설정을 setup 메뉴에서 수행합니다.



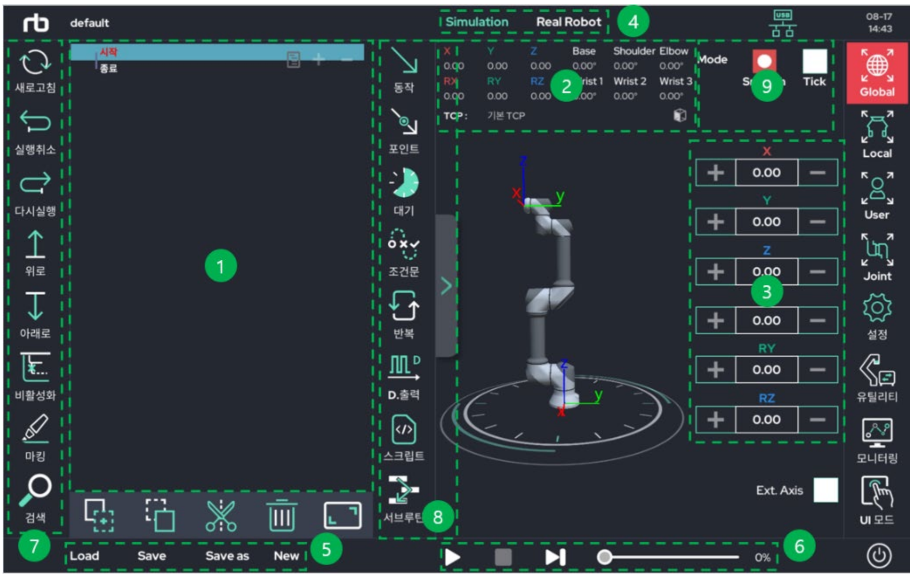
※ 더욱 자세한 내용 설명은 9 장에 설명된 내용을 참고하시기 바랍니다.

제 7 장 프로그래밍 가이드

7.1 아이콘 및 작업 화면 설명

■ Make

화면을 구성하고 있는 요소들을 설명합니다.



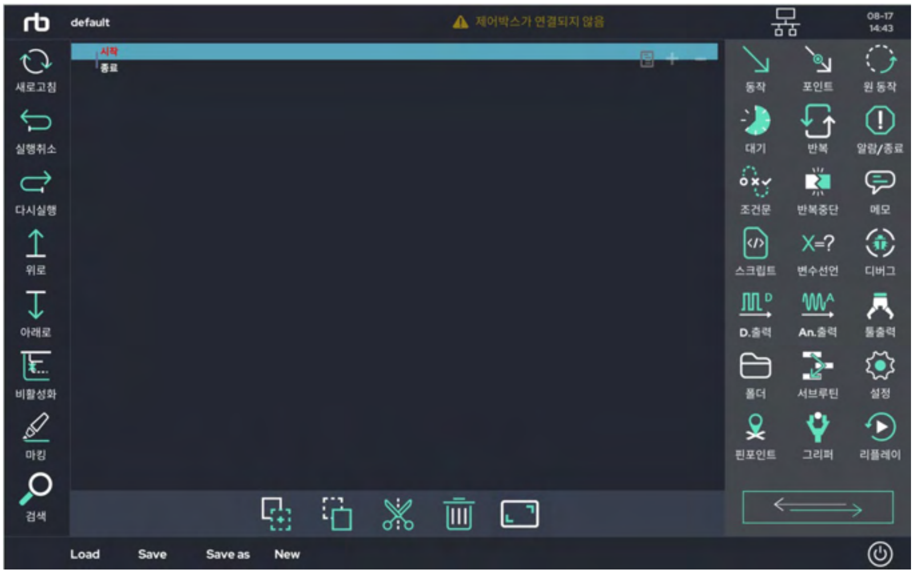
번호	설명
①	프로그램 된 리스트를 트리형식으로 보여줍니다.
②	로봇 팔의 각 관절의 각도와 TCP 의 Cartesian(직교) 좌표 위치를 보여줍니다.
③	TCP 조그: Cartesian 좌표 위치를 변경할 수 있습니다. (베이스/툴/사용자 좌표계 선택 가능) Joint: 조그: 각 조인트의 각도를 변경할 수 있습니다.
④	Simulation 또는 Real 모드로 변환할 수 있는 버튼입니다. ※실제 로봇 팔을 구동하기 위해서는 Real 모드를 선택하여야 합니다.
⑤	저장된 프로젝트를 불러올 수 있고, 새로운 프로젝트를 생성할 수 있습니다.
⑥	티칭 된 프로그램을 실행 또는 멈출 수 있는 버튼이 있고, 속도조절 바가 존재합니다.
⑦	복사/붙여넣기/주석 등 각종 편집 도구가 위치합니다.
⑧	각종 프로그램 기능 (명령어/액션) 등이 배치되어 있습니다. 우측의 화살표를 클릭하면 더 많은 기능들이 포함되어 있습니다.
⑨	조그의 모드를 Smooth 모드와 Tick 모드 중 선택하는 버튼입니다.

※티칭: 사용자가 원하는 대로 로봇 팔을 움직일 수 있도록 프로그램 하는 것을 뜻함.

※TCP(Tool Center Point): 공구중심점으로 주어진 응용에 대하여 기계접속 좌표계에 대해 정의된 점으로, 말단장치(end-effector)의 원점이 됩니다.

[기본 뷰 모드]

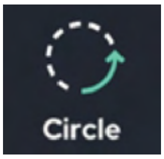
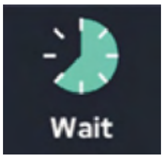
[아이콘 확장 뷰 모드]

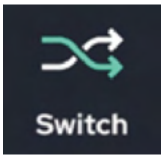
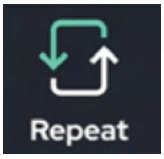
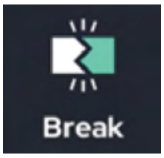
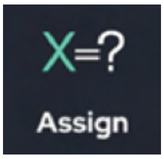
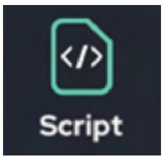
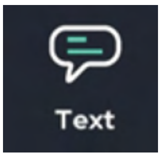

default
제어박스가 연결되지 않음
08-17
14:43
새로고침
실행취소
다시실행
위로
아래로
비활성화
마킹
검색
시작
종료
동작
포인트
원 동작
대기
반복
알람/종료
조건문
반복중단
메모
스크립트
X=?
변수선언
디버그
D.출력
An.출력
툴출력
폴더
서브루틴
설정
핀포인트
그리퍼
리플레이
Load
Save
Save as
New

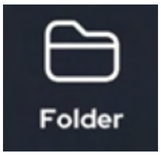
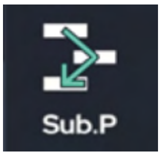
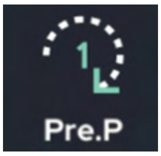
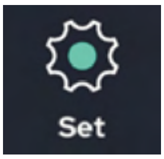
[프로그램 전용 모드]

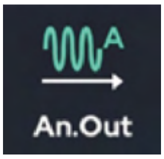
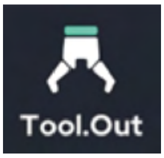
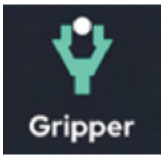
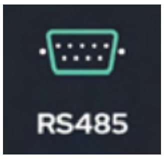
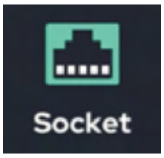
■ **상단 티칭 아이콘(기능 아이콘, Make 화면 상단) 설명**

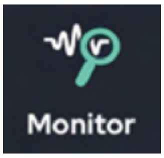
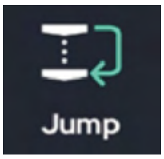
※ 각 기능에 대한 자세한 설명은 뒷부분에 설명되어 있습니다.

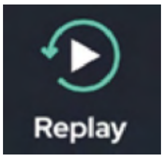
아이콘	설명
	티칭을 위한 로봇 팔의 이동방식을 설정할 수 있습니다. 지정된 목표 관절 각도로 각 관절을 움직이는 MoveJ, 지정된 목표 TCP 값으로 TCP 를 선형적으로 움직이는 MoveL 이 있습니다. 이외에도 두 가지 기본 움직임을 응용할 수 있는 MoveJB, MoveLB, MovePB, MoveJL, MoveITPL 등 총 7 개의 움직임이 내장되어 있습니다.
	Move 의 하위항목으로서 가고자 하는 목표 값을 넣을 수 있는 기능입니다. MoveJ 와 MoveJB 의 경우 목표 값으로 관절의 각도 값을 넣습니다. MoveL, MoveLB, MovePB, MoveJL, 그리고 MoveITPL 의 경우 목표 값으로 TCP 의 직교 좌표 값(x, y, z, rx, ry, rz)을 넣습니다.
	Move 와는 별도로 원형 동작을 위한 기능을 제공합니다. 시작점과 두 점을 찍어서, 3 점을 지나는 호(arc)를 그리는 모드가 존재하고, 중심점과 회전축을 설정하여 원을 그릴 수 있는 모드가 있습니다.
	지정된 조건이나 시간만큼 기다리는 기능입니다. 3 가지 모드가 존재합니다. 지정된 시간만큼 기다리는 기능, 조건이 참일 동안 기다리는 기능, 그리고 조건이 참이면 기다림을 종료하는 기능이 있습니다.
	If 문입니다. 조건에 따라 로봇이 다른 명령을 수행할 수 있도록 분기를 설정할 수 있습니다. If/else if /else 문을 사용할 수 있습니다.

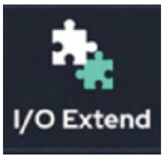
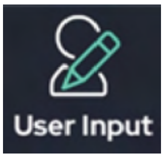
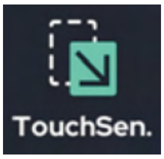
	Switch 문입니다. 조건에 따라 로봇이 다른 명령을 수행할 수 있도록 분기를 설정할 수 있습니다. Switch / case 문을 이용할 수 있습니다.
	반복문입니다. 지정된 조건이나 횟수만큼 하위의 프로그램을 반복합니다. 3 가지 모드가 존재합니다. 지정된 횟수만큼 반복 기능, 지정된 조건이 참일 동안 반복하는 기능, 그리고 조건이 참이면 반복을 끝내는 기능이 있습니다.
	반복문을 강제로 종료하는 기능입니다. 반복문 (repeat 기능) 기능 하위에서 사용되며, 반복문에 특정 조건이 선언되어 있더라도, break 기능이 사용되면 선행적으로 반복문을 탈출합니다.
	프로그램을 종료하는 Halt 기능입니다. 특정 조건이 발생했을 때 프로그램을 종료해야 한다면 이 기능을 사용하면 됩니다.
	변수 선언을 위한 Assign 기능입니다. 변수 형태는 4 타입으로 선언할 수 있습니다. 단일 숫자를 저장하는 variable 타입, 배열 숫자를 저장하는 array 타입, 자세 정보를 저장하는 point 타입, 그리고 문자열을 저장하는 string 타입이 있습니다.
	사용자가 임의의 스크립트를 작성할 수 있는 script 기능입니다. 기능과 별도로 특정 연산/계산이 필요하거나, 변수 치환, 대입 등의 기능을 script 기능을 사용하여 구현 가능합니다.
	프로그램 리스트 트리에 주석과 같이 필요한 부분에 간단한 메모를 할 수 있습니다.

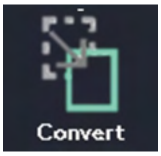
	작성된 프로그램 명령들을 폴더의 하위항목으로 묶어서 모듈로 관리할 수 있도록 도와주는 폴더 기능입니다. 폴더 기능마다 이름을 다르게 부여하여 관리를 도와주는 도구입니다.
	사전에 만들어진 다른 프로그램 파일(티칭 파일)을 현재 프로젝트안에 삽입할 수 있는 기능입니다(서브루틴 기능). 현재 프로젝트에서 Sub.P 기능으로 호출된 하위 프로젝트는 시각적으로 내용을 볼 수는 있으나, 수정은 불가능합니다. 수정이 필요한 경우 해당 프로젝트를 별도로 열어야 합니다.
	하위의 내용을 최초 한 번만 실행하는 Pre-Program 기능입니다. Play 모드와 같이, 프로그램이 계속 돌아가는 경우, Pre-Program 의 하위에 선언된 내용은 최초 한 번만 실행됩니다. 변수 선언이나 통신 연결과 같이, 단발성 명령어의 경우 pre-program 하위에서 관리하면 됩니다.
	쓰레드 기능입니다. 쓰레드 기능의 하위 항목은 메인 프로그램과 병렬로 실행됩니다. 단, 로봇 동작에 관한 명령어는 쓰레드 하위에 넣지 못합니다.
	알람(alarm) 팝업 메시지를 띄울 수 있는 기능입니다. 특정 조건이나 구간에 알람 메시지를 넣어서 사용자의 확인을 요청할 수 있습니다.
	내부 값 디버깅을 위한 Debug 기능입니다. 원하는 변수 값이나 내부 파라미터를 요청하면 팝업 형식으로 해당 값을 확인할 수 있습니다.
	Set 기능입니다. Setup 메뉴에서 세팅하는 각종 설정 파라미터를 프로그램 내부에서 별도로 세팅 가능합니다. 특정 설정 파라미터를 프로그램 실행 중에 변경할 수 있는 기능을 가지고 있습니다.

	컨트롤 박스의 디지털 출력(Digital output)을 제어하는 명령어입니다. 16 개의 포트 중 선택된 포트를 통해 선택된 신호가 출력되는 기능입니다. 각 포트에는 High 신호, Low 신호, Bypass 3 가지를 설정할 수 있습니다.
	컨트롤 박스의 아날로그 출력(Analog output)을 제어하는 명령어입니다. 4 개의 아날로그 포트 중 선택된 포트를 통해 선택된 전압을 출력하는 기능입니다. 각 포트는 0~10 V 의 전압 범주로 출력할 수 있습니다.
	툴 플랜지에는 2 개의 디지털 출력이 있습니다. 2 개의 디지털 출력의 신호를 지정할 수 있습니다. 더불어, 툴 플랜지에서 출력할 전압의 레벨 (0V 또는 12V 또는 24V)을 함께 조절 가능합니다.
	그리퍼 기능입니다. 타사에서 만든 그리퍼 사용에 대한 함수가 미리 내장되어 있습니다. 사용하고자 하는 그리퍼를 선택하고, 해당 그리퍼가 제공하는 기능을 선택하면, 자동으로 손쉽게 그리퍼를 사용할 수 있도록 보조하는 기능입니다.
	툴 플랜지/컨트롤 박스를 통해, RS485/232 출력을 위한 기능입니다. 아스키(Ascii)모드로 출력을 줄 수도 있고, 16 진법(hex) 모드로 출력을 줄 수도 있습니다. 해당 통신 규약은 Setup-serial 메뉴에서 설정합니다.
	소켓 통신을 위한 기능입니다. 소켓을 열어서 특정 서버에 접속하고, 서버에 요청 메시지를 보내고, 서버로부터 특정 데이터를 받아오는 기능을 제공합니다. 소켓 통신은 최대 5 개까지 별도의 서버와 연결 가능합니다.
	모드버스 클라이언트 기능입니다. 특정 IP/주소에 데이터를 요청하고, 반환된 데이터를 가져올 수 있는 기능을 제공합니다. 데이터 요청 주기와 형식 등을 지정 가능합니다. 본 제품에서 제공되는 모드버스 서버에 관련된 프로토콜은 별도로 제공됩니다.

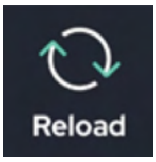
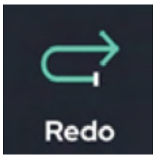
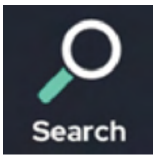
	컨베이어 추적 기능입니다. 컨베이어 속도와 방향에 관한 설정을 하면, 컨베이어 기능의 하위 동작(MoveL, MoveLB, MovePB, MoveITPL, Circle)은 컨베이어 흐름에 더해집니다.
	프로그램 종료 후 마무리 작업으로 수행할 기능을 정의하는 Post Program 기능입니다. 하위에 선언된 기능들은 프로그램 종료 후 순차적으로 수행됩니다. 동작에 관한 명령어는 하위에서 사용하지 못합니다.
	사전에 만들어진 다른 프로그램 파일(티칭 파일)을 현재 문서에 수정 가능한 형태로 삽입하는 기능입니다. Sub.P 의 기능과 유사하지만, Sub.P 의 경우 단순히 해당 프로그램을 호출하는 기능이라면, Template 기능은 현재 프로그램에서 수정 가능한 형태로 다른 프로그램을 복사해서 들고 오는 기능입니다.
	프로그램 구동 중, 실시간으로 관찰하고 싶은 변수 (단일 변수, 배열, 포인트 변수 등)를 선언하는 기능입니다. Monitor 기능에 선언된 변수들은 Make/Play 페이지 우측의 모니터 아이콘을 클릭하면, 프로그램 구동 중 실시간으로 변수 값을 관찰 가능합니다.
	반복 동작을 정의하는 기능입니다. 반복 동작을 수행할 공간에 대한 정보를 정의하고, 각 위치에서 수행할 반복 동작을 정의하면, 모든 공간 지점에서 같은 동작을 수행합니다. 팔래타이징을 이 기능을 통하여 구현 가능합니다.
	특정 자세/위치 정보를 저장하고 변수화 하기위한 핀포인트 기능입니다. 해당 위치로 움직이지 않습니다. 특정 자세의 위치/자세 값을 변수로 저장해 두었다가, 다른 모션 기능에서 참조 (호출)하여 사용 가능합니다.
	프로그램 흐름을 특정 위치/줄 로 점프하여 이동하는 기능입니다. 시작 위치로 돌아가거나, 특정 줄 번호로 프로그램 흐름을 불연속적으로 제어 가능합니다.

	모션 레코딩 기능을 통하여 녹화된 모션을 다시 재생하는 기능입니다. 직접 교시 등을 통하여 녹화된 모션은 이 기능을 통하여 J 나 L 타입으로 다시 재생 가능합니다.
	위빙 동작을 위한 전용 기능입니다. 위빙 동작 하위에 설정된 Move L 이나 Circle 등의 동작은, 설정된 위빙 옵션이 합쳐져서 동작됩니다.
	힘 제어를 위한 기능입니다. 원하는 힘의 작용 방향과 좌표계를 선택합니다. 힘 제어 하위에 포함된 동작들은 힘 제어 기능이 자동으로 부여됩니다.
	아크 용접을 위한 전용 매크로 기능입니다. 일반적인 D.out 이나 Wait 기능등을 통해 구현 가능한 기능을 매크로 형태로 빠르게 사용 가능하도록 묶어둔 전용 기능입니다. 아크 용접기에 관한 설정은 Setup 페이지의 Device 에서 수행합니다.
	TCP 설정 기능은 미리 저장한 TCP 값을 프로그램 실행 중에 불러와 한시적으로 TCP 값을 변경하는 기능입니다. Setup 페이지의 Tool List 에서 TCP 값을 미리 저장할 수 있습니다.
	수동 조작 기능은 프로그램이 실행되는 중에 일시정지하고 직접 교시를 사용할 수 있게 하는 기능입니다.
	G 코드 기능은 지정된 폴더에 G 코드 파일을 위치시킨 경우 사용할 수 있는 기능입니다. 해당 G 코드의 경로를 로봇이 구현하게 됩니다.

	HMI, PLC 등의 타제품을 쉽게 사용할 수 있도록 기능을 내장한 기능입니다. 사용하고자 하는 제품을 선택 가능하고 선택한 제품을 사용하기 위한 세부 기능을 선택할 수 있습니다.
	I/O 확장 모듈을 추가하였을 때 사용 가능한 기능입니다. I/O 확장 모듈의 디지털/아날로그 출력을 설정할 수 있으며 설정 방식은 기존의 D.출력, An.출력과 동일합니다.
	유저 입력 기능은 프로그램 실행 중 일시정지 하여 변수/배열/포인트/문자열/Global/ROM 의 값을 사용자가 입력하여 변경 가능하도록 하는 기능입니다. 사용자의 상황에 따라 값을 변경할 수도, 무시하고 건너뛰는 것도 가능합니다.
	용접 어플리케이션에 사용하기 위한 기능입니다. 모재의 이동을 감지하여 이동 및 이동 방향을 반영해 용접을 진행할 수 있도록 해줍니다.
	프로그램의 시작 위치 또는 로봇의 관절 0 점 위치로 로봇을 이동시키는 기능입니다. 이동 방식은 MoveJ, MoveL 중 선택 가능합니다.
	디지털 용접기를 사용하기 위한 기능입니다. 사용하고자 하는 브랜드의 디지털 용접기를 선택한 후 모드, 옵션을 선택하면 디지털 용접기를 쉽게 사용할 수 있습니다.
	프로그램과 동시에 실행되는 일반/비정지 쓰레드와 달리 메인 프로그램에서 호출하면 실행되는 이벤트 일반/비정지 쓰레드를 호출하는 기능입니다.

	메인 프로그램을 다른 프로젝트로 바꾸고자 할 때 사용하는 기능입니다.
	로봇 관절과 부가축을 동시에 동작 시키기 위한 기능입니다.
	로봇 이외에 부가축(추가적인 모터)를 제어하기 위한 기능입니다. 최대 6 개까지 추가 가능합니다.

■ 편집 아이콘 설명 (Make 화면 좌측)

아이콘	설명
	현재 열려 있는 파일을 다시 불러옵니다. ※Save 를 누르지 않고 Reload 를 누를 경우 마지막으로 Save 된 파일을 불어오게 되니 주의하시기 바랍니다.
	작업을 한 단계 이전으로 되돌립니다. 최대 50 회까지 가능합니다.
	작업을 한단계 이후로 재실행 합니다. 최대 50 회까지 가능합니다.
	명령어를 한 칸씩 위로 올릴 수 있습니다
	명령어를 한 칸씩 밑으로 내릴 수 있습니다.
	주석화 (억제) 기능입니다. 선택된 명령어를 실행하지 않도록 만듭니다. 주석화가 된 명령어는 프로그램에 존재하지만 실행되지 않습니다.
	원하는 프로그램 라인에 하이라이트 (마킹)을 표시할 수 있습니다. 중요한 프로그램 라인에 밑줄을 그리는 기능입니다.
	작성한 프로그램 내에서 사용한 기능에 대한 검색 기능입니다. 단, 영어로만 검색 가능합니다.

	선택된 명령어를 복사합니다. 복사한 명령어는 다른 위치에 붙여 넣을 수 있습니다.
	복사 또는 잘라낸 명령어를 선택한 위치에 붙여 넣습니다.
	선택된 명령어를 잘라냅니다. 잘라낸 명령어는 다른 위치에 붙여 넣을 수 있습니다.
	선택된 명령어를 삭제합니다.
	편집 모드를 확대 모드로 변경하는 아이콘입니다.

■ 로봇 팔 움직임 및 기타 기능 (Make 화면 우측)

Icon	Description
	베이스에 고정된 Global 좌표계를 기준으로 TCP 의 위치를 움직일 수 있습니다.
	TCP 에 고정된 Local 좌표계 (툴 좌표계)를 기준으로 TCP 의 위치를 움직일 수 있습니다.
	사용자 정의 좌표계 (User Coordinate)를 기준으로 TCP 의 위치를 움직일 수 있습니다.
	로봇 팔의 각 관절을 따로 움직일 수 있습니다.
	자주 사용하는 특수 기능들이 모여 있습니다. 시스템의 입출력 상태 정보, 유저 좌표계 (user-coordinate) 설정 정보, 로봇의 전류/온도 정보 등 기타 상태 및 설정 값을 볼 수 있습니다.
	유저 좌표계 설정, 자동 TCP 찾기 기능 등, 조그와 함께 사용하면 편리한 설정을 별도로 모아두었습니다. 해당 설정들은 기본적으로 Setup 메뉴에서도 설정 가능합니다.
	Monitor 기능을 통해 선택된 변수들의 값을 실시간으로 관찰하는 창입니다. 선택된 변수뿐만 아니라, 자주 확인해야 되는 시스템 변수도 함께 표시됩니다.
	UI 모드를 선택할 수 있는 기능입니다. 사용자의 수준, 환경에 맞게 UI 모드를 선택할 수 있습니다.

■ 시스템 기능 버튼

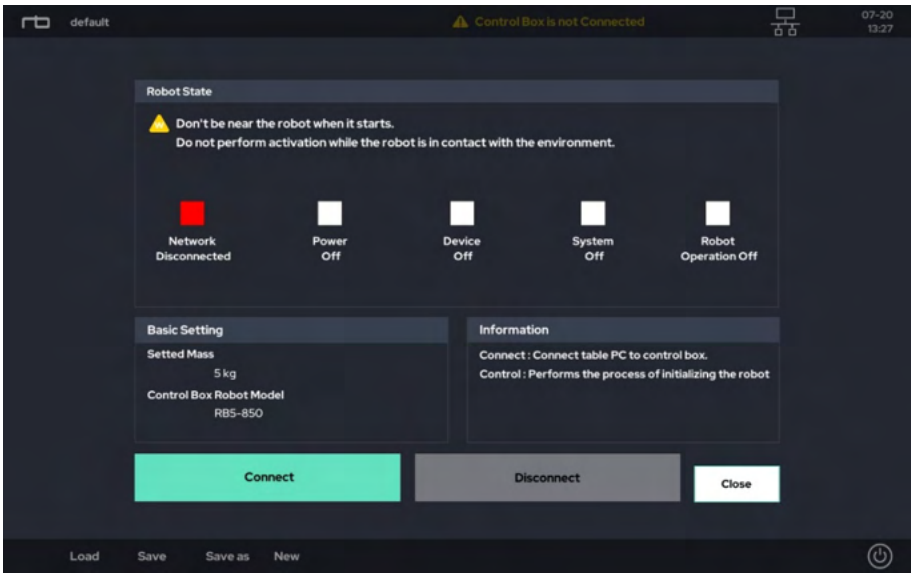
아이콘	설명
	홈 화면 및 다른 페이지로 이동하는 버튼입니다. (좌측 상단에 배치되어 있음).
	UI 가 종료됩니다. 만약 로봇이 활성화되어 있고, UI 태블릿 PC 와 연결이 되어 있는 경우, 로봇의 전원도 함께 내려갑니다. 스크린 락(lock) 기능이 있습니다(우측 하단에 배치되어 있음).

7.2 티칭 환경 만들기

로봇 티칭(프로그래밍)은 Make 화면에서만 가능합니다. Play 혹은 Setup 화면이 띄워져 있다면 UI 상단 부분에 있는 버튼을 통해 Make 화면으로 이동하시기 바랍니다. 혹은 Home 화면으로 이동하여 Make 화면으로 이동도 가능합니다.

■ 태블릿 PC 와 컨트롤 박스 연결하기

아이콘을 통해 로봇 컨트롤 박스와 태블릿 PC 를 연결해야만 로봇을 티칭할 수 있는 환경이 만들어집니다. State 아이콘은 각 화면의 좌측 하단에 배치되어 있습니다.
State 아이콘을 클릭하면 다음과 같은 화면이 나타나게 됩니다.



Robot State 화면이 띄워지게 되면 Connect(연결) 버튼을 눌러서 컨트롤 박스와 태블릿 PC 를 연결합니다.

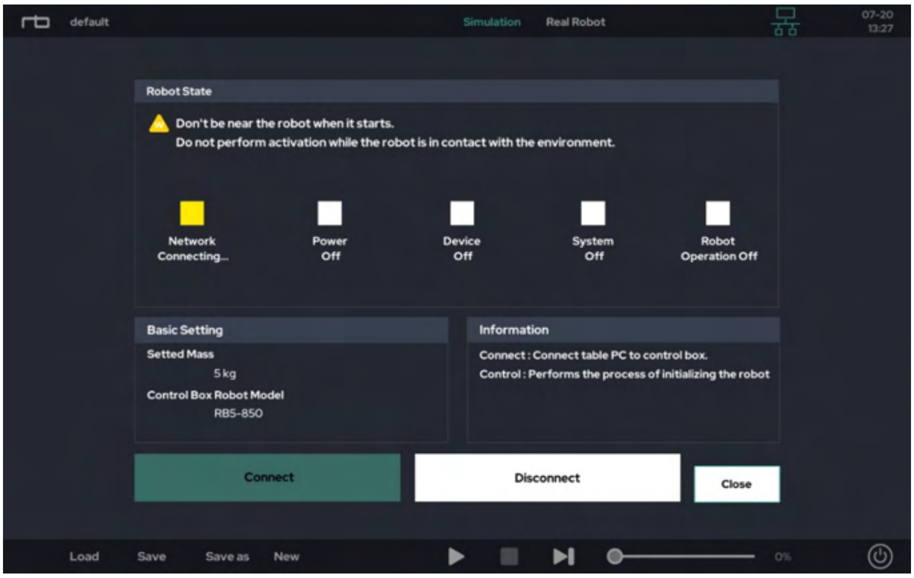
- **Connect 버튼:** 컨트롤 박스와 ↔ 태블릿 PC 사이의 통신을 연결합니다.

주의

주의:

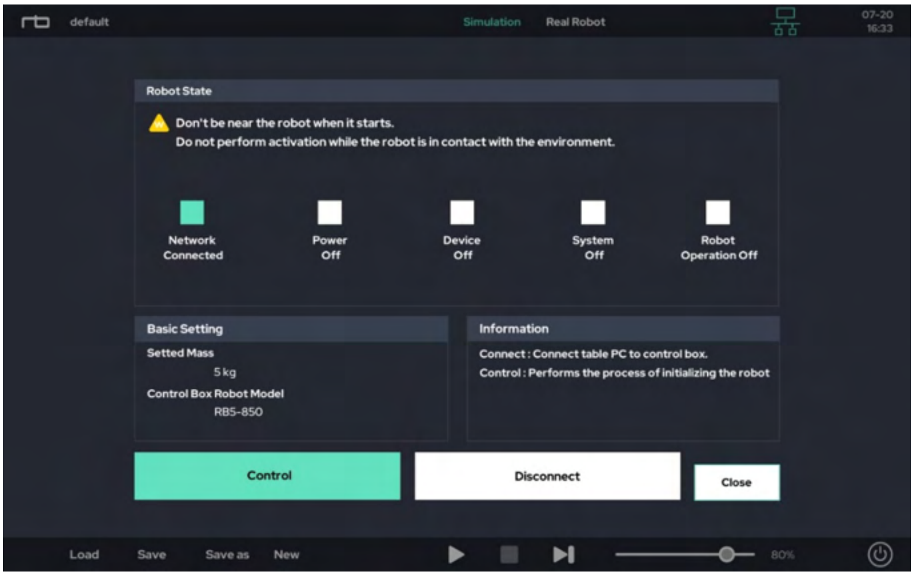
1) 컨트롤 박스의 전원을 켰는지 확인하시고, 비상 정지 스위치도 해제되어 있는지 먼저 확인 후에 연결을 시도하십시오. 컨트롤 박스가 켜져 있지 않을 경우 Device Off 에 빨간 불이 켜지고 위와 같은 화면으로 되돌아 가게 됩니다.

컨트롤 박스와 컴퓨터가 연결 중 일 때는 아래와 같은 화면이 나타나게 됩니다.



연결을 시도 중에는 첫번째 상태 등이 노란색으로 표시됩니다.

연결이 완료되면 아래와 같은 화면이 나타나게 되고, Control 이라는 버튼이 활성화가 됩니다.



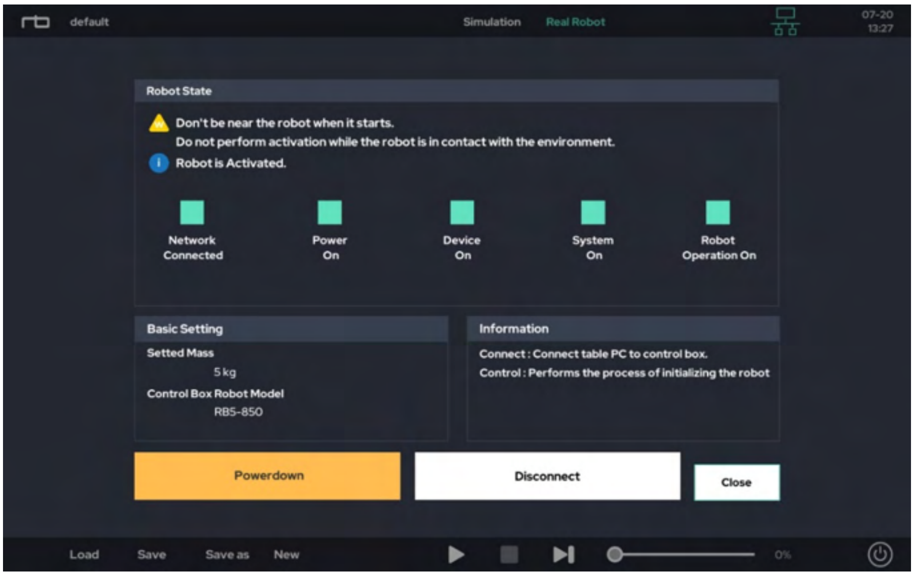
컨트롤 박스와 연결을 마치고 난 다음에는 로봇 컨트롤 시스템을 켜야 합니다.

로봇은 활성화된 Control 버튼을 누르면 켜지게 됩니다.

- **Control 버튼:** 로봇 팔을 구동 가능 상태로 초기화 합니다.

Control 버튼을 누르게 되면 로봇 팔을 제어 가능한 상태로 초기화를 합니다. 이 때 관절의 브레이크가 해제되면서 로봇 팔에서 순차적으로 소리가 나게 됩니다.

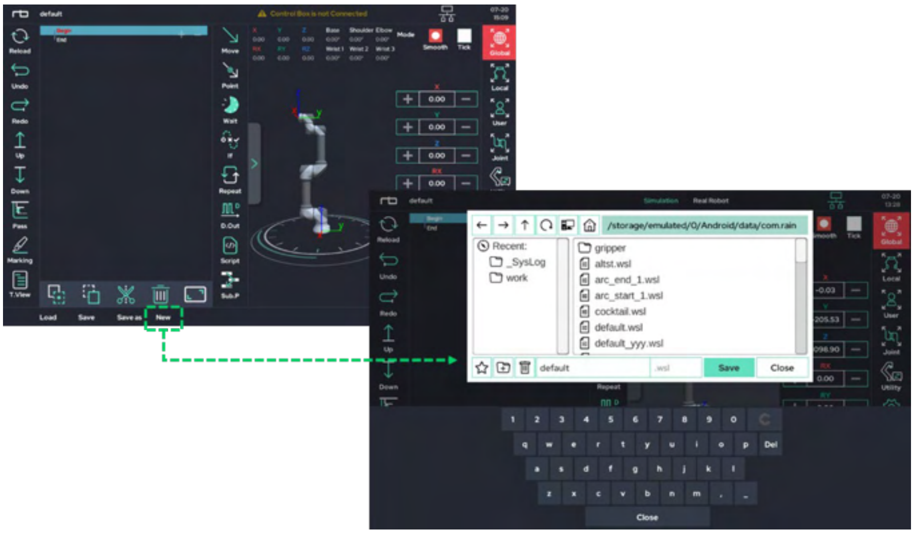
모든 시스템이 다 켜져서 연결이 되면 아래 화면과 같이, 모든 상태 표시 등에 파란불이 켜지게 됩니다.



※ 만약 Robot Operation On 에 파란불이 들어오지 않고, 빨간 불이 켜지게 되면 비상정지 스위치 개폐 여부 확인, 툴 부하 세팅 확인, 로봇 설치 각도 확인 등, 안내 팝업 메시지에 따라 설정 및 운용 상황을 확인해 보십시오.

■ 새로운 프로젝트 생성하기

새로운 프로젝트를 생성하기 위해서는 Make 화면 하단 부분에 있는 New 버튼을 누르게 되면 파일 이름을 부여할 수 있는 화면이 나타나게 됩니다.



새로 생성되는 모든 프로젝트의 기본 이름은 default 로 되어 있습니다. 이름을 변경하기 위해 기본 이름이 적힌 박스를 선택하면 그림의 화면처럼 가상 키보드가 나타나면서 이름을 변경할 수 있게 됩니다. 이름을 변경한 후에는 반드시 Save 버튼을 눌러야 프로젝트가 생성됩니다.

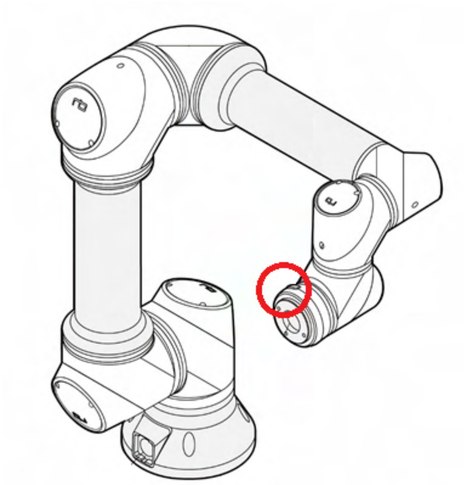
7.3 티칭 하기(사용자 프로그램 만들기)

■ **로봇 움직이기**

제공하는 로봇 구동 방식은 다음의 2 가지입니다.

- **직접 교시:** 로봇 팔에 있는 버튼을 눌러서 사용하는 직접 교시 기능.
- **조그 사용:** UI 상의 조그를 사용하여 로봇 구동.

■ **직접 교시(Direct-teaching / gravity compensation)**



로봇의 말단 툴 플랜지에는 직접 교시를 위한 실리콘 버튼이 있습니다. 직접 교시를 누르고 있는 동안, 사용자는 로봇의 자세를 자유롭게 변형할 수 있습니다.

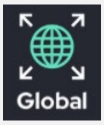
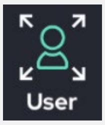
주의

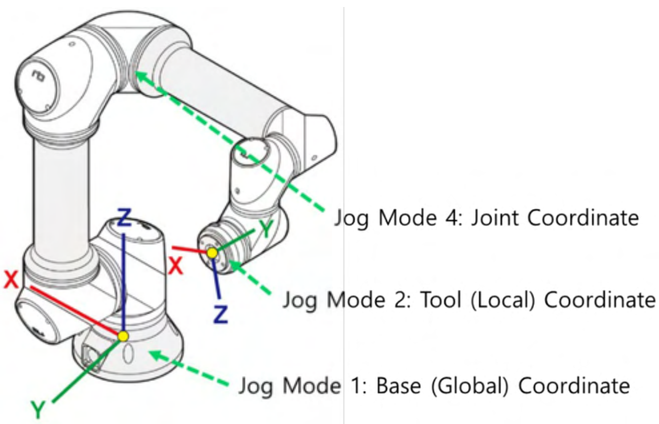
주의:

1) 직접 교시 기능은 로봇이 활성화되어 있을 때만 사용 가능합니다.
2) 툴 플랜지에 부하(load)가 장착되어 있을 경우, 부하 세팅을 사전에 하지 않는다면, 직접 교시 기능이 오작동 할 수 있습니다. 부하 세팅은 setup-tool 화면에서 합니다.
3) 직접 교시 기능의 관절 별 감도 설정은 setup-interface 에서 조절 가능합니다.
4) 로봇이 완전히 정지되어 있는지 확인 후 직접 교시 기능을 사용하십시오.

■ 조그 사용

조그는 위의 그림과 같이 4 가지 모드가 있습니다.

Mode 1		직교 좌표계 움직임 (베이스 좌표계 기준)
Mode 2		직교 좌표계 움직임 (로컬 툴 좌표계 기준)
Mode 3		직교 좌표계 움직임(사용자 지정 좌표계 기준)
Mode 4		개별 관절 움직임

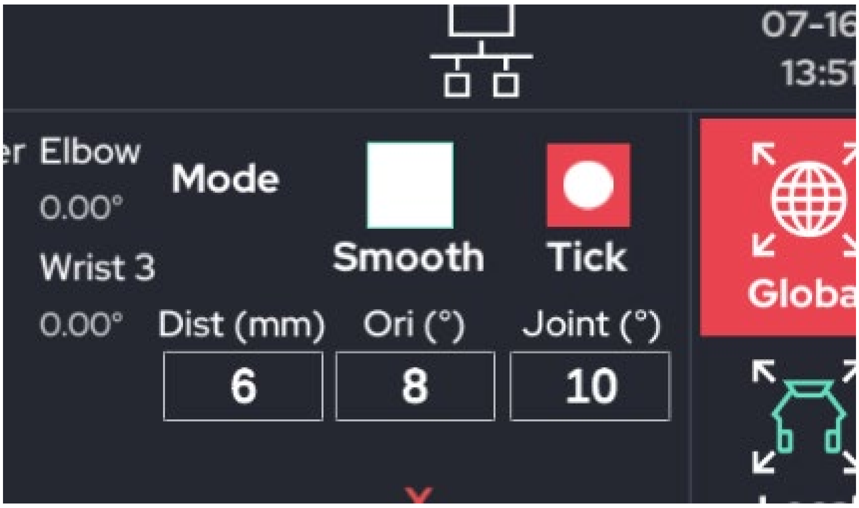


조그의 운영 방식은 2 가지입니다.

- Smooth: + - 버튼을 누르는 동안 연속적으로 움직입니다.
- Tick: 지정된 거리나 각 만큼 단계별로 움직입니다.

※ 위의 두 가지 운영 방식에 대한 선택은 Make 화면 우측 상단의 토글 버튼을 사용하여 선택 가능합니다.

※ Tick 을 선택하고 조그 사용시, 움직이는 거리나 각도는 Setup-Interface 에서 설정 가능합니다. 또는 아래와 같이 실시간 팝업 창에서 값을 지정 가능합니다.

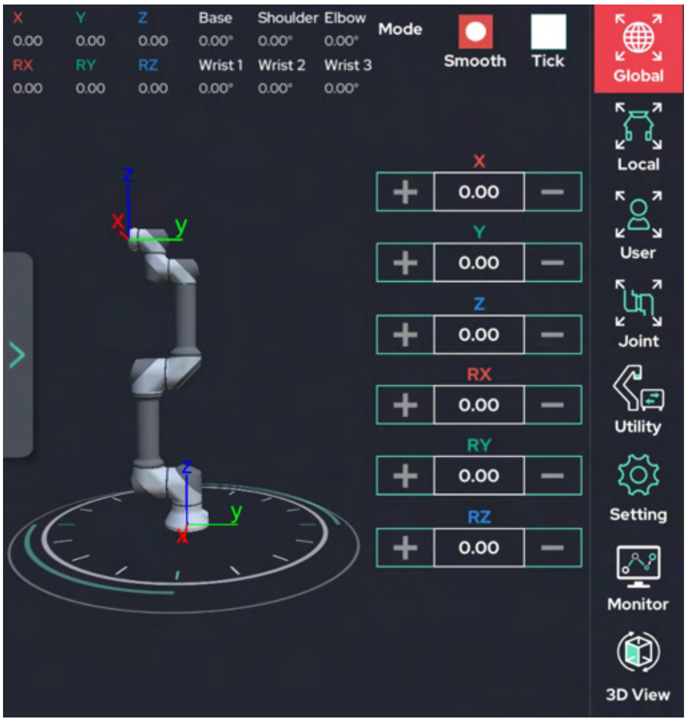


경고

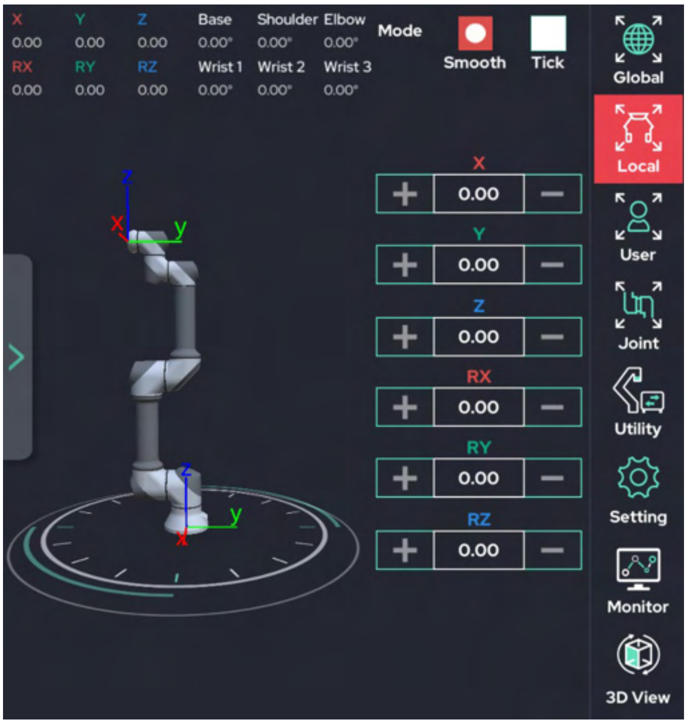
경고:

1) 로봇을 조그를 사용해 움직일 때는 주변에 장애물이나 사람이 없는지 확인 후 작동하시기 바랍니다.
2) Setup-Interface 에서 안전 슬라이더 기능을 활용하여 좀 더 안전한 환경에서 조그를 운영하는 것을 추천합니다(공장 출하 시 안전 슬라이더 기능은 활성화되어 있습니다).

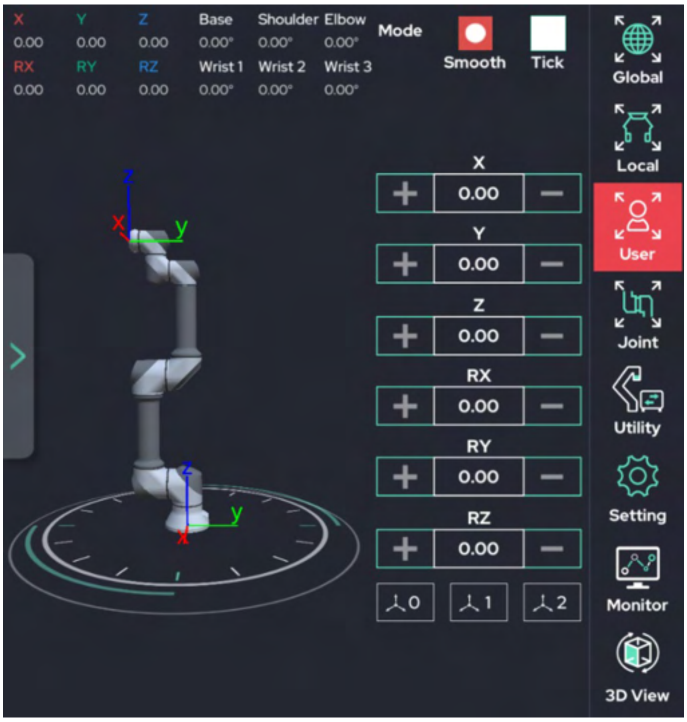
[조그 타입 1: 베이스 좌표계 기준 TCP 조그]



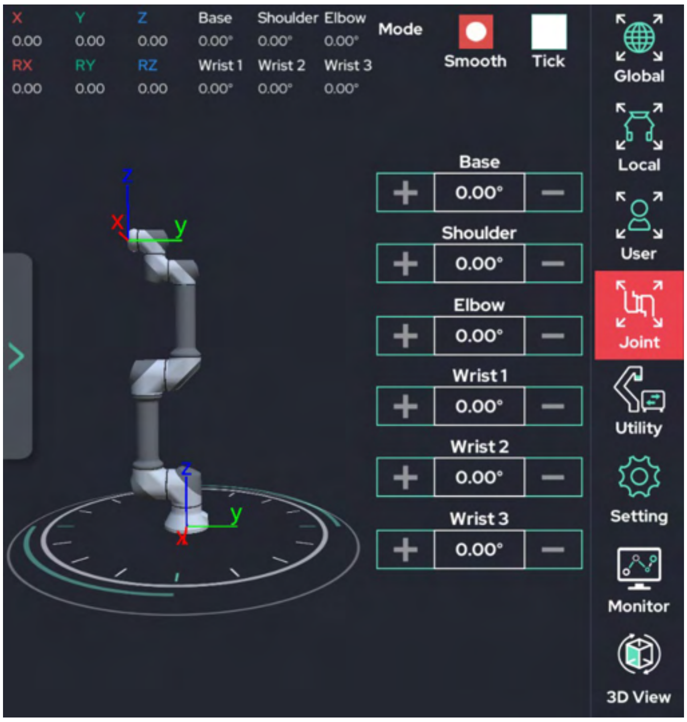
[조그 타입 2: 로컬/툴 좌표계 기준 TCP 조그]



[조그 타입 3: 사용자 좌표계 기준 TCP 조그]



[조그 타입 4: 개별 관절 좌표계 기준 조그]



■ 리얼 모드와 시뮬레이션 모드

로봇 움직임을 테스트 또는 구동할 때, 2 가지의 모드가 제공됩니다.

- **시뮬레이션 (Simulation) 모드:**

  실제 로봇을 움직이지 않고, UI 화면 상에서 가상으로 로봇 팔을 구동 시켜보는 모드입니다.

  새로운 동작을 티칭하였을 때는 안전상, 시뮬레이션 모드를 먼저 실행해보는 것을 권장합니다.

- **리얼 로봇 (Real Robot)모드:**

  UI 화면 상의 로봇은 물론, 실제 로봇을 움직이는 모드입니다.

주의:

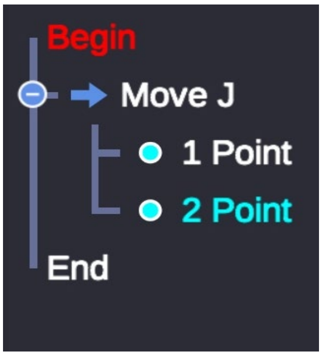
1) 리얼 모드로 로봇 구동은, 로봇이 활성화되어 있을 때만 사용 가능합니다.
2) 시뮬레이션 모드는 로봇을 활성화하지 않고도, 컨트롤 박스와 태블릿 UI 가 연결만 되어 있다면 사용 가능합니다.
3) 리얼 모드로 로봇 구동 시, 실제 로봇이 움직입니다. 따라서 주변 환경이 안전한지 확인하는 과정을 거친 후 사용하세요.

■ 로봇 동작 티칭하기

가장 기본적인 로봇 동작 티칭은 Move **Move** 기능과 Point **Point** 기능이 담당합니다. 두 아이콘 모두 **Make** 화면 상단 바에 위치합니다.

- **Move** 기능: 로봇 팔의 동작 속성을 정의합니다.
- **Point** 기능: **Move** 의 하위 항목으로써, 움직임을 통해 도착할 자세 또는 위치를 설정하는 부분입니다.

이 두 기능을 사용하여 프로그램을 구성하면 전체적인 구성은 아래와 같이 됩니다.

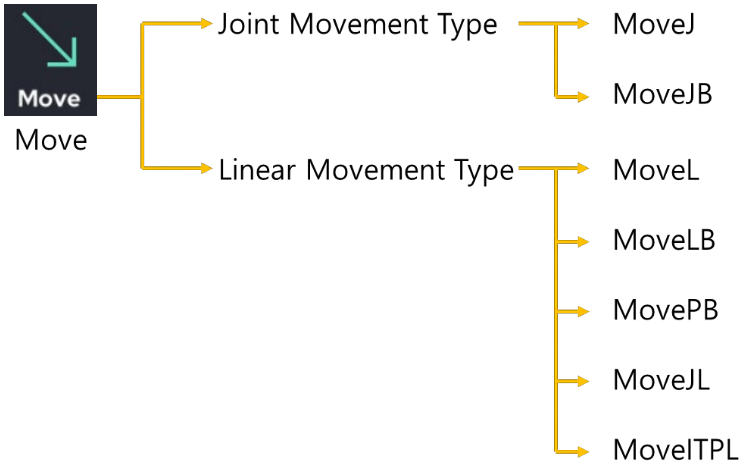


← 동작 속성

← 첫 번째 목표 지점
← 두 번째 목표 지점

Move 기능과 **Point** 기능 각각에 대한 자세한 설명은 아래 페이지에서 이어집니다.

■ Move 기능



Move 기능은 로봇 팔을 동작 속성, 동작 옵션을 정하는 기능입니다. Move 기능은 하위에 **Joint** 계열과 **Linear** 계열로 구성됩니다.

① Joint 계열(Joint Movement Type)

Joint 계열은, 로봇의 말단 (TCP) 움직임과는 별개로 각 관절의 각도를 목표 값으로 움직이는 동작 속성입니다. 하위 요소로 MoveJ 와 MoveJB 가 있습니다.

▷ **MoveJ (Move Joint):**

각 관절의 각도를 목표포인트의 값으로 변경시킵니다. 구동시간이 가장 많이 소요되는 관절에 맞추어 다른 관절의 구동속도가 조정됩니다.

▷ **MoveJB (Move Joint Blend):**

시작 포인트를 출발하여 각 포인트 사이를 정지없이 Move J 방식으로 부드럽게 이동합니다.

② Linear 계열(Linear Movement Type)

Linear 계열은, 로봇 팔의 말단 (TCP)을 직교 좌표계 기준으로 움직이는 동작 속성입니다. 따라서 움직임의 목표 값으로 직교 좌표계 값을 사용합니다. 하위 요소로 MoveL, MoveLB, MovePB, MoveJL, 그리고 MoveITPL 가 있습니다.

▷ MoveL (Move Linear):

TCP 가 시작 포인트(현재위치)로부터 목표 포인트(설정위치)까지 직선으로 이동하는 방식입니다. 툴의 orientation 은 가장 적은 이동각도를 가지도록 회전합니다. TCP 의 이동과 orientation 의 변화 중 많은 시간이 소요되는 작업에 맞추어 다른 작업의 구동속도가 조정됩니다.

▷ MoveLB (Move Linear Blend):

시작 포인트(현재위치)를 출발하여 각 포인트 사이를 정지없이 Move L 방식으로 부드럽게 이동합니다.

각 포인트마다 Blend Radius 를 지정하여 전후경로를 해당 거리만큼 잘라내고 원호로 이은 경로를 따라 이동합니다. 지정된 Radius 값이 전후경로의 길이의 절반보다 클 경우 두 경로 중 짧은 경로길이의 절반길이로 자동으로 수정됩니다.

Move LB 에는 두 가지 모드가 있습니다. Constant 모드는 툴의 orientation 을 시작 포인트(현재위치)의 값을 유지하며 이동하는 모드이고 Intended 모드는 각 포인트에 저장된 orientation 을 따라 변화하는 모드입니다.

▷ MovePB (Move Point Blend):

MoveLB 와 유사하지만, 더 범용적으로 사용할 수 있는 기능입니다. 각 중간 점 (via point)마다, blend 정도를 거리 단위로 설정할 수도 있고, 백분율 단위 (%)로도 설정할 수 있습니다. 더불어, 각 중간 점마다 속도를 별도로 설정할 수 있습니다.

▷ MoveJL (Move J with Linear Input):

MoveL 과 같이 목표 지점의 직교 좌표계 값을 입력으로 사용합니다. 다만, 해당 지점까지 직선으로 가지 않고, MoveJ 의 방식을 활용하여 갑니다. 직교 좌표계 입력이 들어오면, 이를 역기구학을 통하여 목표 관절 각도로 변환시키고, 이를 다시 MoveJ 에 입력하는 방식을 사용합니다.

▷ MoveITPL (Move Interpolation):

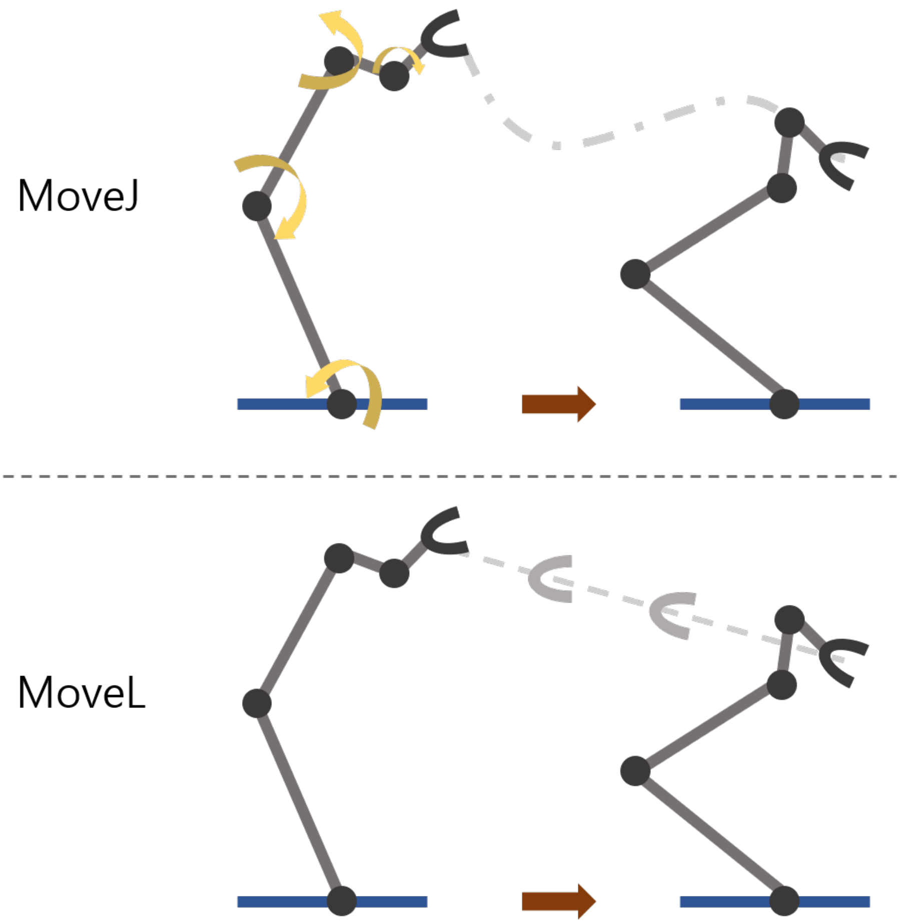
시작포인트(현재위치)를 출발하여 각 포인트 사이를 정지없이 Move L 방식으로 부드럽게 이동합니다.

MoveLB 나 PB 는 각 경유지를 지나지 않고 블랜드되어 이동하지만, MoveITPL 은 각 경유지를 정확히 지나는 궤적으로 움직입니다. 따라서 별도의 블랜드 세팅이 존재하지 않습니다.

MoveITPL 에는 두 가지 모드가 있습니다. Constant 모드는 툴의 orientation 을 시작포인트(현재위치)의 값을 유지하며 이동하는 모드이고 Intended 모드는 각 포인트에 저장된 orientation 을 따라 변화하는 모드입니다.

각 중간 경유점 마다 속도를 별도로 설정할 수 있습니다.

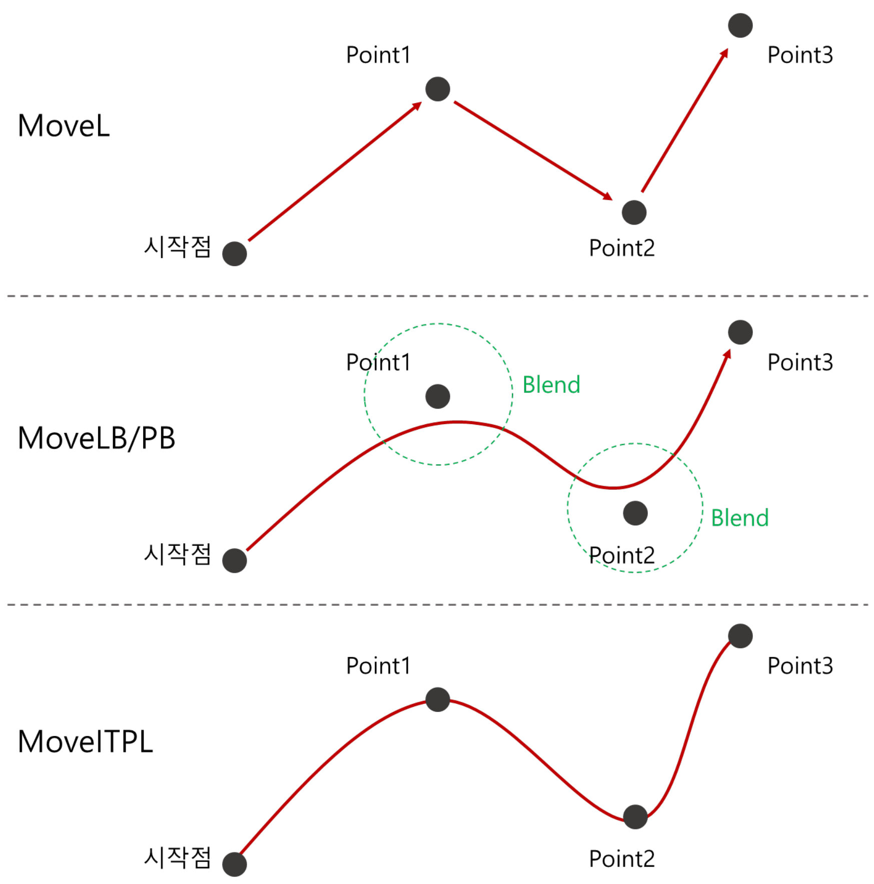
■ MoveJ 와 MoveL 의 차이



MoveJ 는 말단(TCP)의 움직임 궤적은 고려하지 않고, 출발 지점의 관절 각도와 목표 지점의 관절 각도 정보만을 이용한 동작 모드입니다. 구동시간이 가장 많이 소요되는 관절에 맞추어 다른 관절의 구동속도가 조정됩니다. 따라서 목표 지점 값으로 6 개 관절의 관절 각도를 입력 받습니다.

MoveL 은 말단(TCP)의 궤적을 출발 지점에서부터 목표지점까지 선형으로 움직이도록 역기구학(inverse kinematics)을 이용하여 움직이는 동작 모드입니다. 따라서 목표 지점 값으로 직교 좌표계 값 6 개(x, y, z, Rx, Ry, Rz)를 입력 받습니다.

■ MoveL, MoveLB/PB, MoveITPL 의 차이



MoveL 의 경우 시작점과 도착점 사이를 일직선으로 선형적인 경로로 이동합니다. 도착점을 정확하게 도달하지만 도착점에서는 일시적으로 움직임 속도가 0 이 됩니다.

MoveLB/PB 의 경우 도착점을 제외한 나머지 점들은 경유지로 이용하며, 지정된 왜곡률(blend rate)나 왜곡거리(blend distance)에 따라 경유지를 정확히 지나지는 않지만, 멈추지 않고 지나갑니다.

MoveITPL 의 경우 도착점을 제외한 나머지 점들은 경유지로 이동하며, 경유지를 정확히 통과하는 궤적을 생성합니다. 생성된 궤적을 멈추지 않고 지나가며, 경유지 마다 별도의 속도 세팅이 가능합니다.

주의

주의:

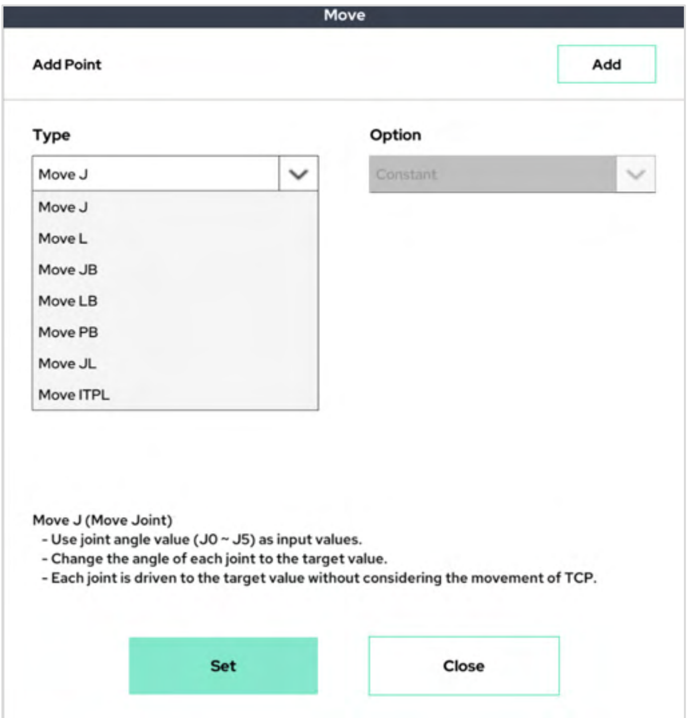
1) MoveL, MoveLB, MovePB, MoveJL, MoveITPL 5 가지 동작 모드는 역기구학 연산을 통해 로봇을 움직입니다. 따라서 역기구학 연산이 불가능한 특이점 영역 (Singular position)에서는 움직임이 제한될 수 있습니다.

2) 또한 로봇의 데드존 영역(Dead-Zone) 또는 그 근처에서는 특정 관절이 예상보다 빠르게 움직이거나 움직임이 제한될 수 있습니다. 데드존 영역에 관한 정보는 1.7 장에 서술되어 있습니다.

■ Move 의 동작 속성 변경

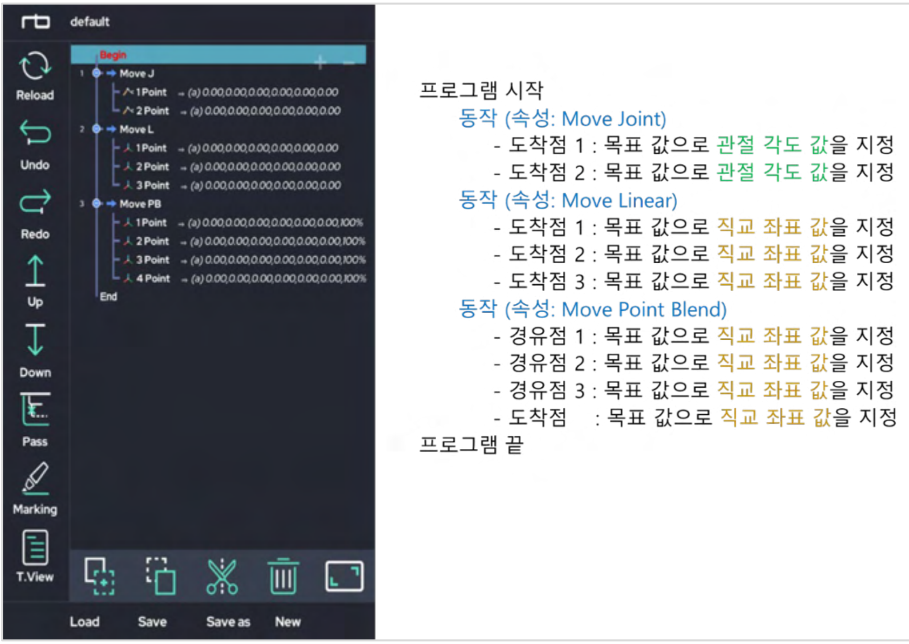
최초로 Move 기능을 프로그램에 넣게 되면 아래의 그림과 같이 Move 기능이 프로그램 tree 안에 생성이 됩니다. 최초 생성시 기본으로 MoveJ 옵션으로 생성됩니다.

위에서 설명된 Move 의 다른 옵션으로 선택하기 위해서는, 생성된 MoveJ 부분을 클릭하면 아래와 같은 팝업창이 뜹니다.



원하는 동작 속성/옵션을 선택하고, 설정을 하면 Move 의 동작 속성이 변경됩니다.

위의 기능들을 활용하여 동작을 티칭 시 프로그램 예시는 아래와 같습니다.



MoveJ, MoveJB

관절을 기준으로 움직이는 동작 속성이기 때문에 하위 항목인 Point 에는 이동하고자 하는 목표 자세의 관절 각도 값이 설정됩니다.

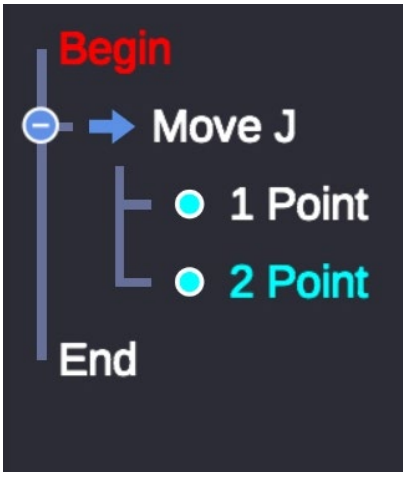
※ 로봇 암은 6 개의 관절로 구성되어 있기 때문에, MoveJ, JB 의 하위 Point 에는 6 개 관절 값에 관한 정보가 설정됩니다.

MoveL, MoveLB, MovePB, MoveJL, MoveITPL

직교 좌표계 값을 기준으로 또는 목표로 움직이기 때문에 하위 항목인 Point 에는 이동하고자, 또는 지나가고자 하는 목표 지점의 직교 좌표계 값이 설정됩니다.

※ 직교좌표계는 6 개의 값(x, y, z, Rx, Ry, Rz)로 구성되어 있기 때문에, MoveL, LB, PB, JL, ITPL 의 하위 Point 에는 6 개의 직교 좌표계 값에 관한 정보가 설정됩니다.

■ Point 기능



← 동작 속성

← 첫 번째 목표 지점

← 두 번째 목표 지점

앞서 Move 기능에서 설명하였듯, Point 기능은 Move 기능의 하위 기능입니다. Move 기능이 동작의 속성을 지정하고, Point 기능은 각 속성에 따라 로봇 팔을 움직일 목표 지점 정보를 설정하는 기능을 담당합니다.

Point 기능에서는 Move 기능에서 설정한 동작 속성에 따라 설정하는 목표 값이 달라지게 됩니다.

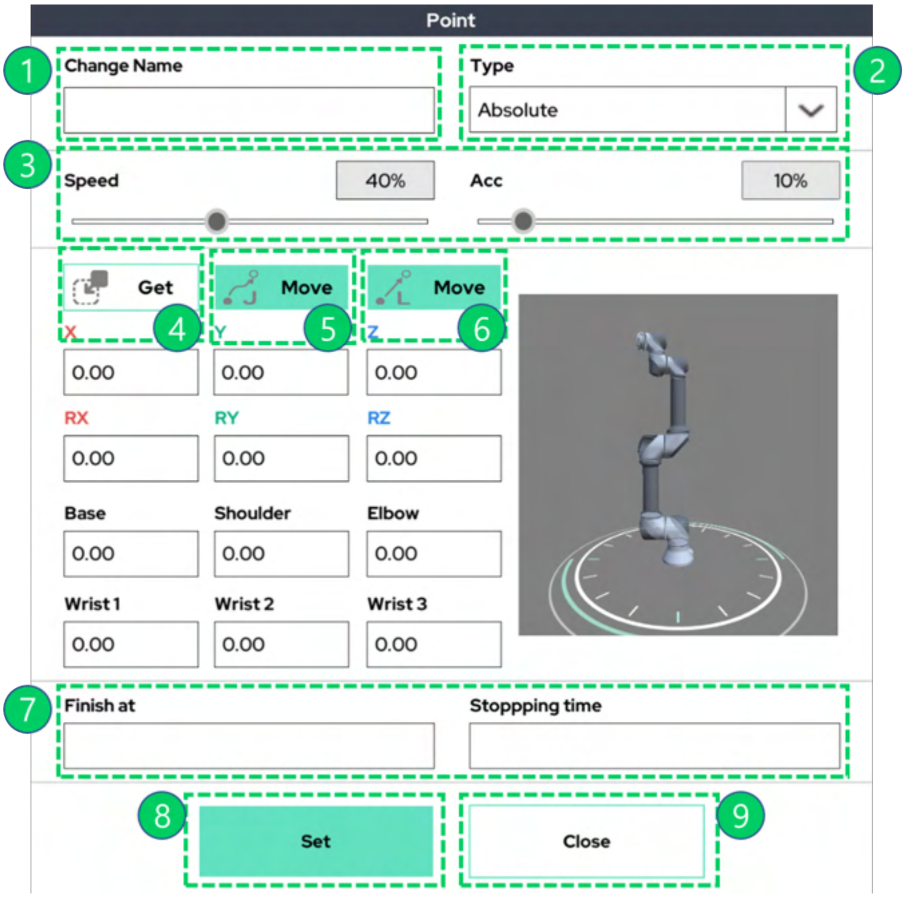
▶ Joint Movement Type(MoveJ, MoveJB)의 하위 Point:

이동하고자 하는 목표 자세의 6 개 관절 각도 값이 설정

▶ Linear Movement Type(MoveL, MoveLB, MovePB, MoveJL, MoveITPL)의 하위 Point:

이동하고자, 또는 지나가고자 하는 목표 지점의 직교 좌표계 값이 설정

Point 기능의 설정 팝업 창의 기본 구성은 아래와 같습니다.



각 영역에 대한 설명은 아래의 표와 같습니다.

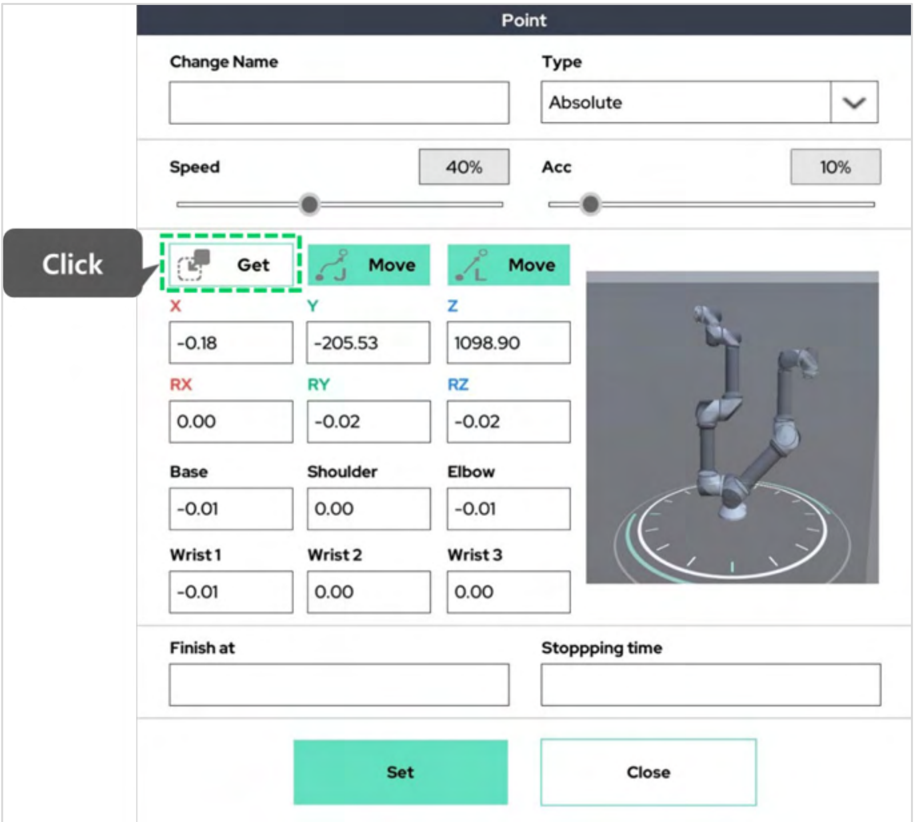
	설명
①	포인트의 이름을 설정할 수 있습니다. 이름 설정은 필수 입력은 아닙니다. 이름을 설정하면, 이후에 해당 Point 의 위치 정보를 변수처럼 사용 가능합니다.
②	Point 의 설정 타입을 고를 수 있는 옵션입니다. Joint Move 속성의 하위 point 는 3 가지 설정 옵션이 있고, Linear Move 속성의 하위 point 는 4 가지 설정 옵션이 존재합니다. Point 최초 생성시 기본 타입은 'Absolute' 입니다.
③	목표 지점까지 움직일 속도와 가속도를 설정합니다.
④	현재 로봇 자세/각도에 관한 정보를 획득하여, 정보로 가져옵니다. 조그 또는 직접 교시를 통해 로봇을 원하는 지점/자세로 이동 후, Get (가져오기)을 누르면 해당 지점의 정보가 Point 정보에 반영됩니다. 새롭게 반영된 정보를 저장하려면 Set 버튼 (7 번)을 눌러야 합니다. Point 타입(2 번)에 따라 이 버튼이 지원되는 타입이 있고, 지원되지 않는 타입도 존재합니다.
⑤	지정된 자세로 이동하는 기능입니다. 버튼을 누르고 있는 동안, 저장된 자세로 이동합니다. 이동은 Joint movement 타입으로 이동합니다. 이동이 완료되면, 이동이 완료되었음을 알려주는 팝업 메시지가 뜹니다. Point 타입(2 번)에 따라 이 버튼이 지원되는 타입이 있고, 지원되지 않는 타입도 존재합니다.
⑥	지정된 자세로 이동하는 기능입니다. 버튼을 누르고 있는 동안, 저장된 자세로 이동합니다. 이동은 Linear Movement 타입으로 이동합니다. 이동이 완료되면, 이동이 완료되었음을 알려주는 팝업 메시지가 뜹니다. Point 타입(2 번)에 따라 이 버튼이 지원되는 타입이 있고, 지원되지 않는 타입도 존재합니다.

⑦	동작의 탈출 조건(Finish at 에 입력)과 탈출 시간(Stopping time 에 입력)을 지정합니다. 필수 입력 항목이 아닙니다. 이 항목을 입력을 하지 않을 경우, 동작은 정상적으로 목표 지점/자세에 도달 후 종료됩니다. 이 항목에 특정 조건과 탈출 시간을 설정할 경우, 목표 지점/자세에 도달하기 전이라도, 탈출 조건을 만족하게 되면 탈출 시간에 맞추어 동작을 멈추게 되고 다음 동작을 이어갑니다. 탈출 시간의 최소값은 0 초입니다.
⑧	변경된 설정을 저장합니다.
⑨	설정 창을 닫습니다. 변경된 설정이 있다면 Set 버튼(7 번)을 눌러야 합니다.

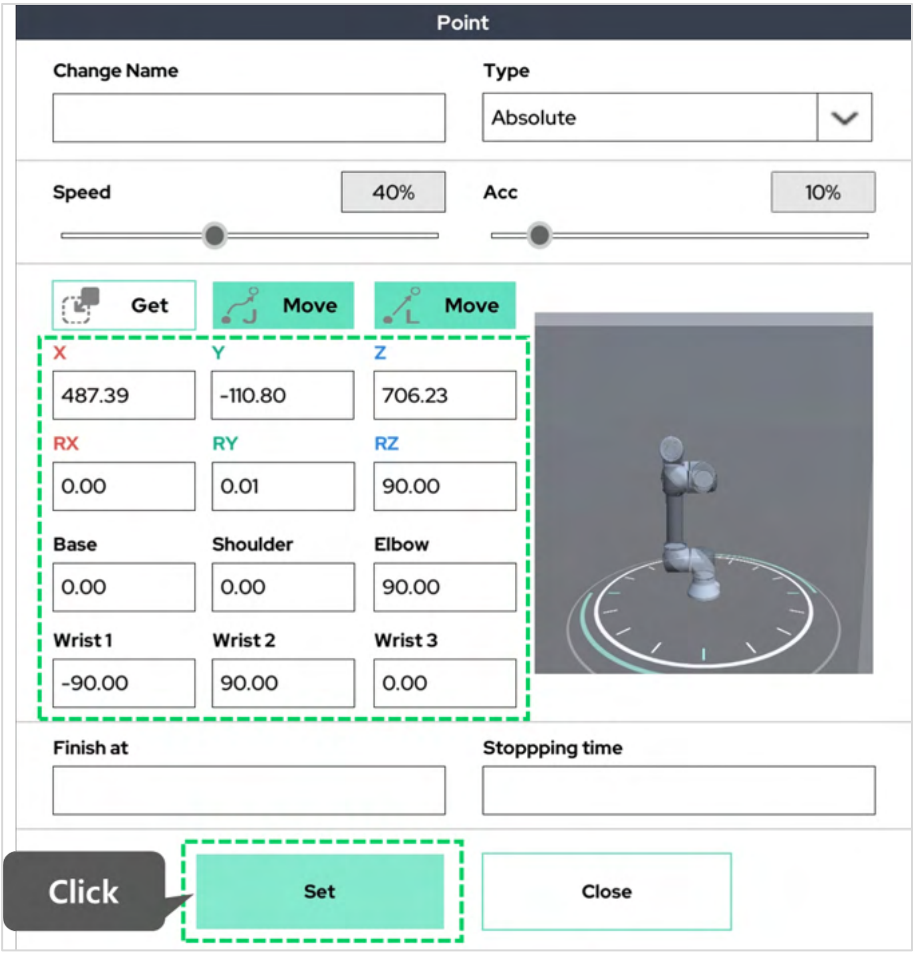
※ 위의 표 4 번 항목 Get(가져오기)에서 제공되는 기능을 도식화하면 아래와 같습니다.

1. 조그 / 직접 교시 기능을 활용해 원하는 자세/위치로 이동

2. Get 버튼을 눌러서 현재 자세/위치 정보 가져오기



3. 반영 확인 후 저장



※ 위의 표 6 번 항목 Finish at/Stopping time 에서 제공되는 기능을 도식화하면 아래와 같습니다.

■ Finish at 기능을 사용하지 않을 경우

(빈 칸으로 남겨둘 경우)

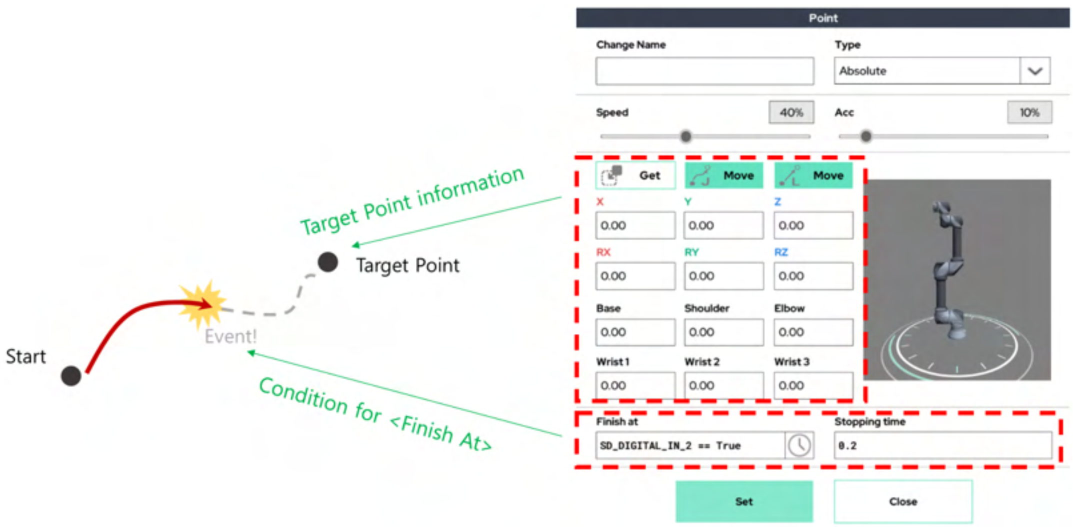
원래 설정된 목표 지점까지 도착 후 동작 종료, 다음 명령 실행

■ Finish at 기능을 사용할 경우

(특정 조건식을 기입할 경우)

목표점까지 도착하지 못해도, Finish at 조건 발생시 동작종료, 다음 명령 실행

동작 중 조건이 발생하지 않는다면, 정상적으로 목표점 도착 후 다음 명령 실행



Move 의 각 타입에 따라 아래와 같은 설정 옵션이 존재합니다.

Joint Move 타입 하위 Point		
옵션	Absolute	>목표 관절 각도 값으로, 고정된 값을 사용합니다. >조그/직접 교시로 로봇을 원하는 자세로 만들고, Get (가져오기) 기능을 통해 관절 각도 값을 설정할 수 있습니다.
	Variable	>사용자가 자유롭게 목표 관절 각도를 입력할 수 있습니다. >조그/직접 교시로 로봇을 원하는 자세로 만들고, Get (가져오기) 기능을 통해 관절 각도 값을 입력할 수 있습니다. >변수 및 수학 연산을 활용하여 입력 가능합니다.
	Relative	>이전 각도 자세에서 변화를 주고 싶은 각도 변화량을 설정합니다. >모든 관절 변화량에 0 을 넣는다면, 이전 자세와 동일한 각도 자세로 움직이라는 명령이 됩니다. >변수 및 수학 연산을 활용하여 입력 가능합니다.

>> 계속

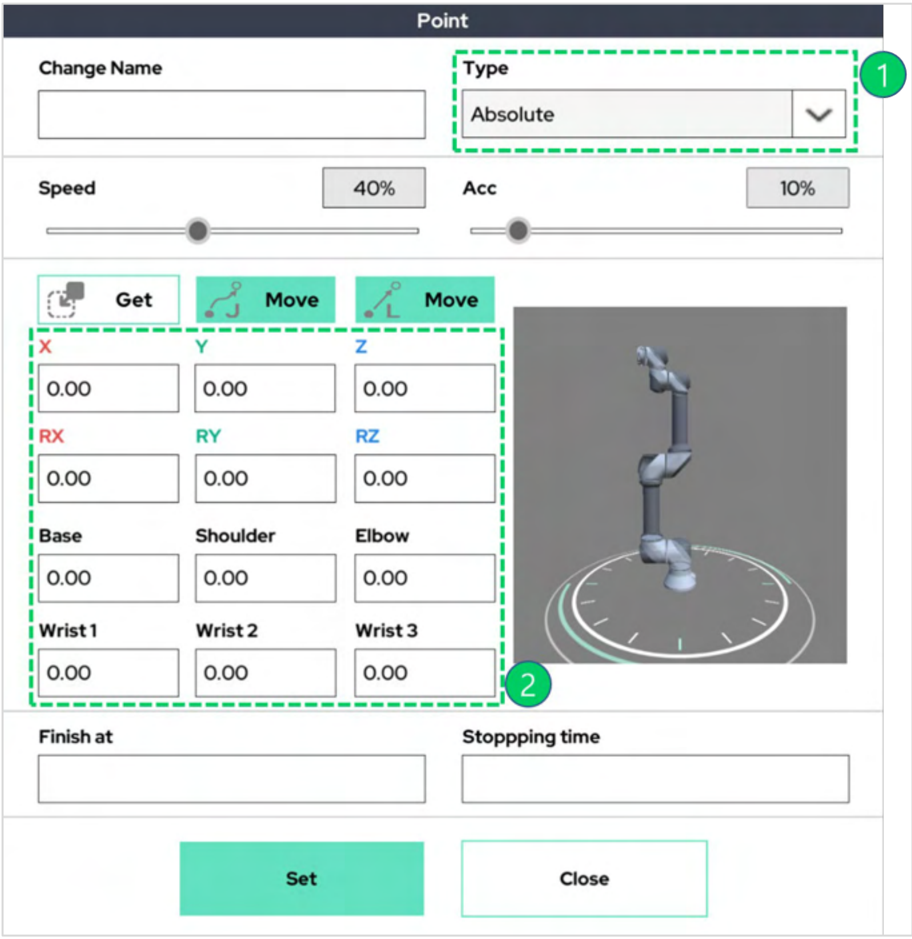
<table>
<tr><th colspan="3">Linear Move 타입 하위 Point</th></tr>
<tr><td rowspan="3">옵션</td><td>Absolute</td><td>>목표 직교 좌표 값으로, 고정된 값을 사용합니다.
>조그/직접 교시로 로봇의 TCP 를 이동 후, Get(가져오기) 기능을 통해 직교 좌표 값을 설정할 수 있습니다.
>설정한 직교 좌표 값의 기준 좌표계는 로봇 팔의 Base 좌표계입니다 (제조사 설정 기본 좌표계).</td></tr>
<tr><td>Variable</td><td>>사용자가 자유롭게 목표 직교 좌표 값을 입력할 수 있습니다.
>설정한 직교 좌표 값의 기준 좌표계는 로봇 팔의 Base 좌표계입니다 (제조사 설정 기본 좌표계).
>조그/직접 교시로 로봇의 TCP 를 이동 후, Get(가져오기) 기능을 통해 직교 좌표 값을 설정할 수 있습니다.
>변수 및 수학 연산을 활용하여 입력 가능합니다.</td></tr>
<tr><td>Relative</td><td>>이전 지점 또는 특정 point 로부터 상대적인 거리/각도 (offset)를 설정하여 목표지점을 지정할 수 있습니다.
>Reference Point (기준점)에서는 상대적으로 이동할 지점을 고르는 기능입니다. 기본 값은 PT_LAST_TCP 로써 직전 도착 지점을 나타냅니다. 이외에도 임의의 point 를 기준으로 할 수도 있습니다.
>Reference Frame(기준좌표계)에서는 어떤 좌표계를 기준으로 상대이동을 지정할 것인지를 정할 수 있습니다. 기본 값은 Frame_Base 로써 로봇 팔의 Base 좌표계를 나타냅니다. 유저 좌표계나, 툴의 local 좌표계로 변경 가능합니다.

>변수 및 수학 연산을 활용하여 입력 가능합니다.</td></tr>
</table>

	User Coordinate	>Variable 과 비슷하지만, 유저 설정 좌표계를 기준으로 목표 지점을 설정하는 기능입니다. >Reference Frame (기준좌표계) 설정을 통해, 기준으로 사용하고 싶은 유저좌표계를 선택 가능합니다. >원하는 기준 좌표계를 선택하고, Get (가져오기) 기능을 활용하면, 현재 로봇의 자세/위치 정보가 선택된 좌표계 기준으로 자동 입력할 수 있습니다. >예를 들어 유저좌표계 0 번을 선택하고, 모든 직교 좌표 값에 0 을 입력 시, 유저 좌표계 원점으로 TCP 가 이동합니다. >변수 및 수학 연산을 활용하여 입력 가능합니다.

각 Point 타입에 따른 UI 구성은 아래와 같습니다.

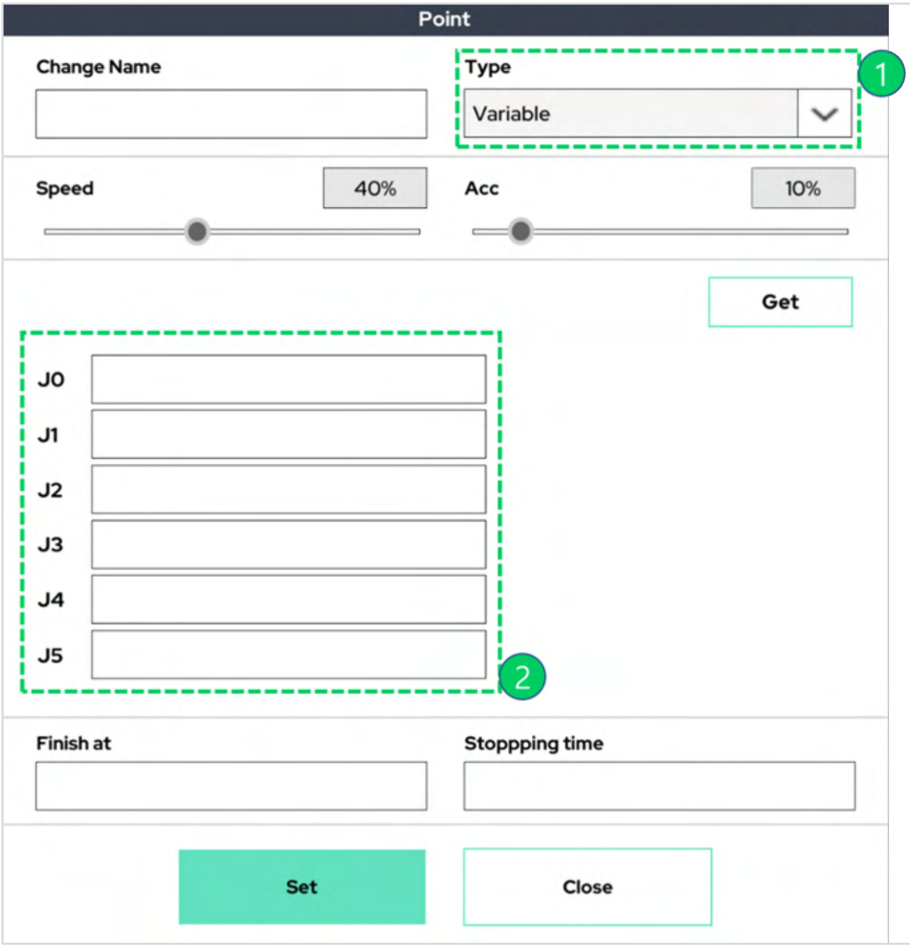
▶ Joint 타입 - Absolute 옵션



① Absolute 옵션입니다.

② Get (가져오기) 버튼을 통해 로봇의 자세/각도 값이 저장됩니다.

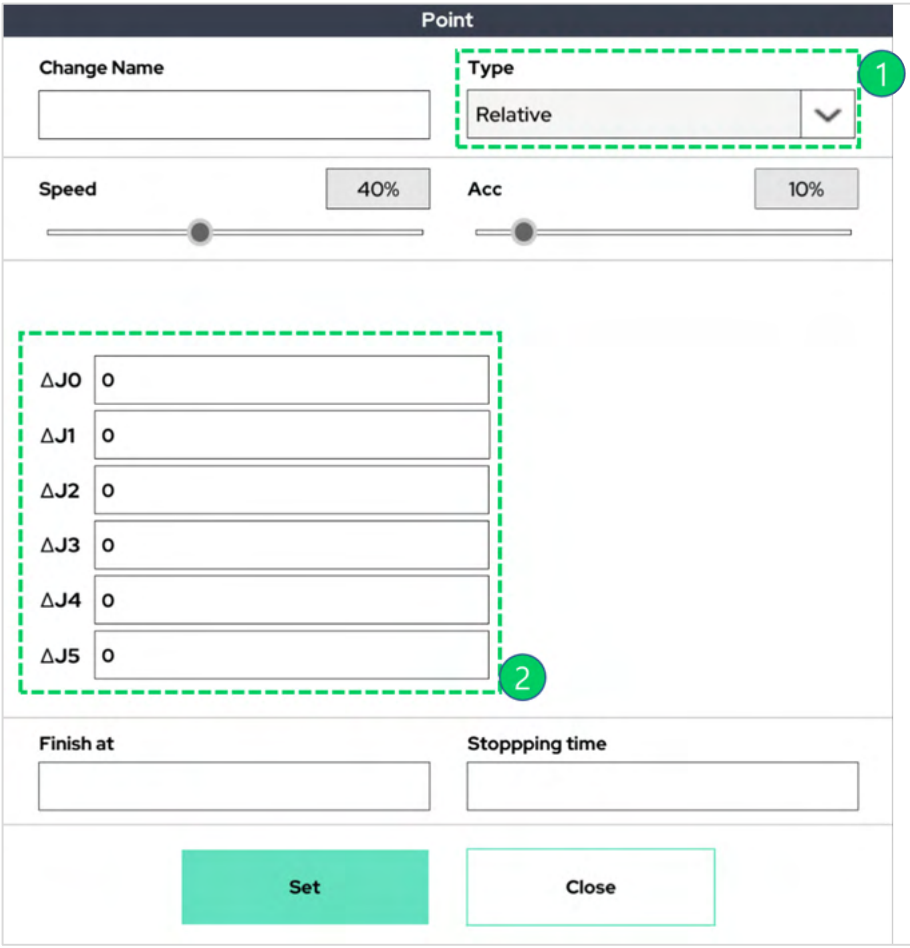
▶ Joint 타입 - Variable



① Variable 옵션입니다.

② 목표 자세에 대한 관절 각도를 사용자가 직접 입력하거나, 변수화 된 정보들을 수식으로 입력할 수 있습니다.

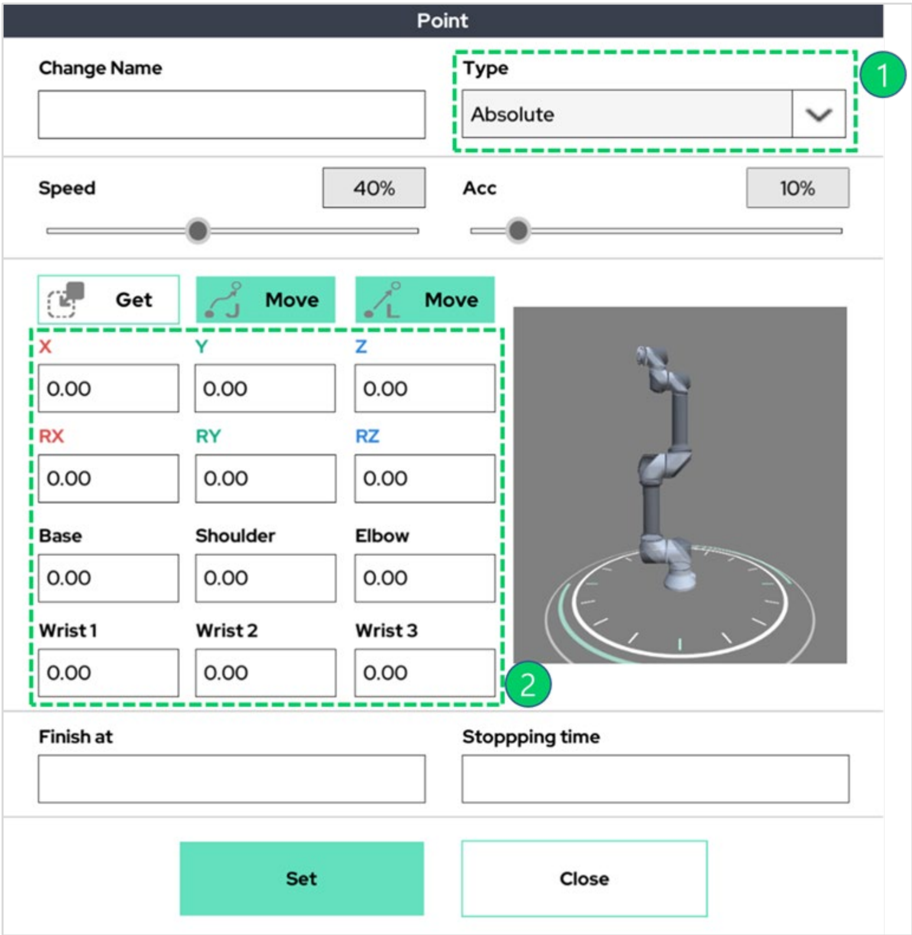
▶ Joint 타입 - Relative 옵션



① Relative 옵션입니다.

② 이전 관절 각도에서 각 관절을 얼마나 상대적으로 움직일지 각도 단위로 입력합니다. 변수화 된 정보나 수식을 입력할 수 있습니다.

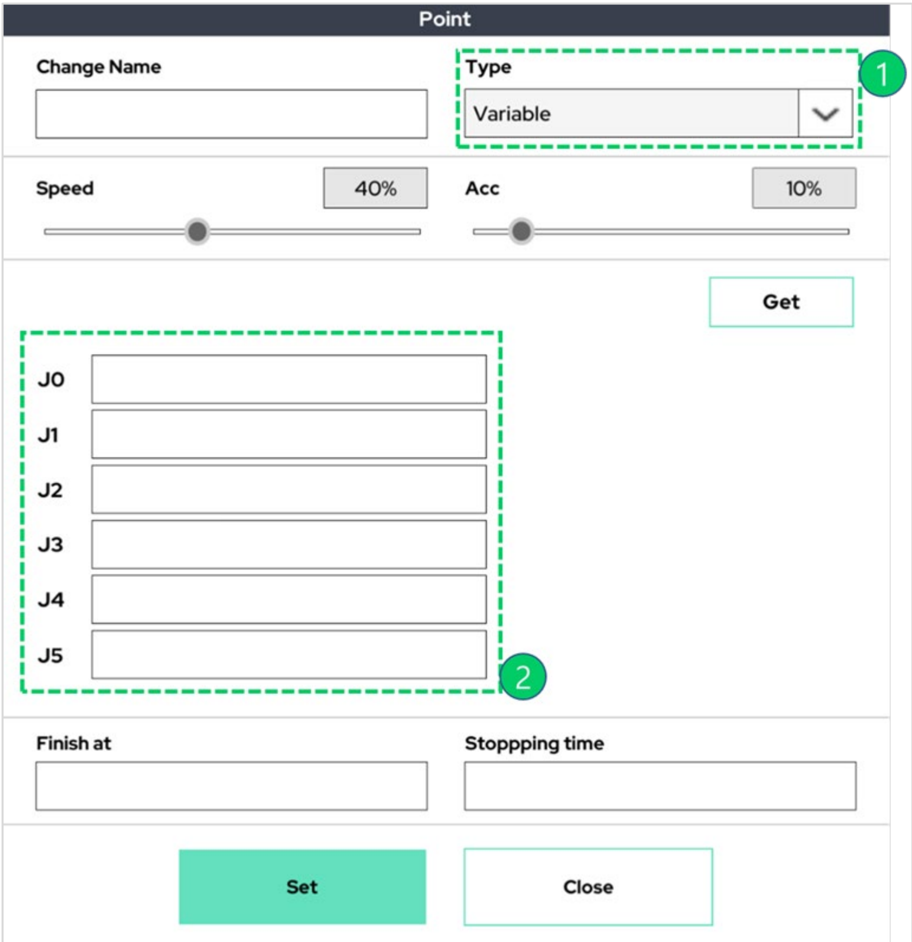
▶ Linear 타입 - Absolute 옵션



① Absolute 옵션입니다.

② Get (가져오기) 버튼을 통해 로봇의 자세/위치 값이 저장됩니다. 직교 좌표계 값의 기준 좌표계는 로봇 베이스 좌표계입니다.

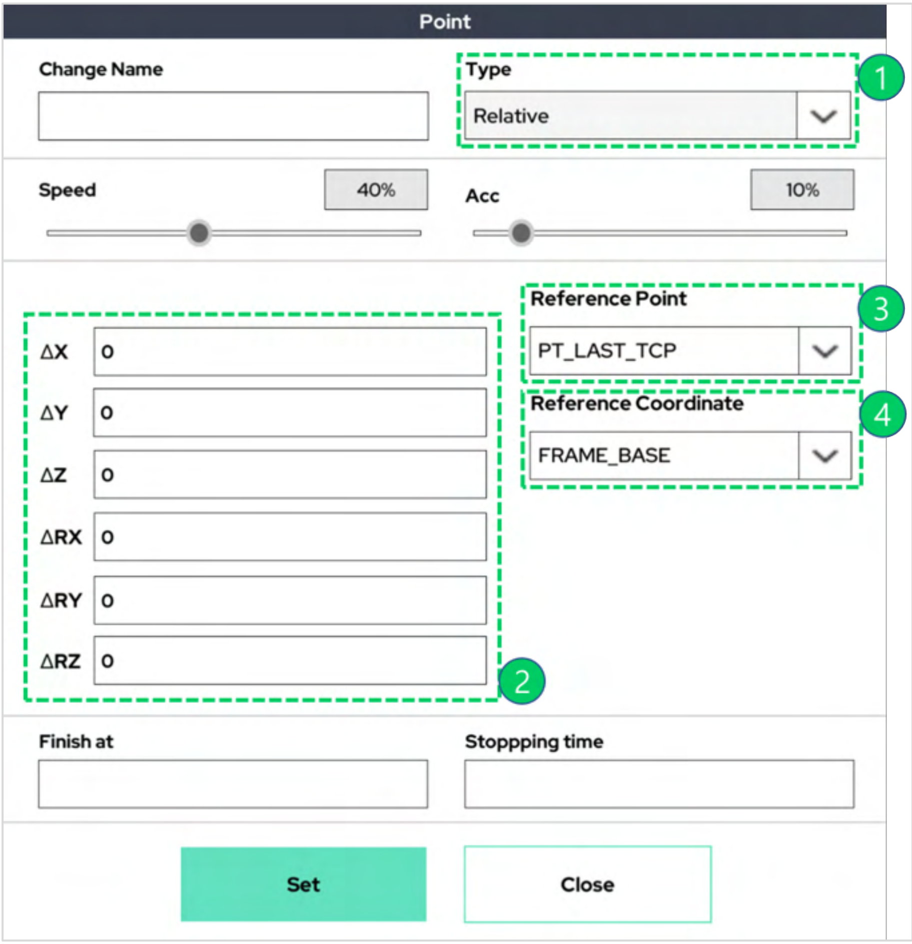
▶ Linear 타입 / Variable 옵션



① Variable 옵션입니다.

② 사용자가 자유롭게 목표 직교 좌표 값을 입력할 수 있습니다. 변수화 된 정보들을 수식으로 입력할 수도 있습니다. 설정한 직교 좌표 값의 기준 좌표계는 로봇 팔의 베이스 좌표계입니다.

▶ Linear 타입 / Relative 옵션



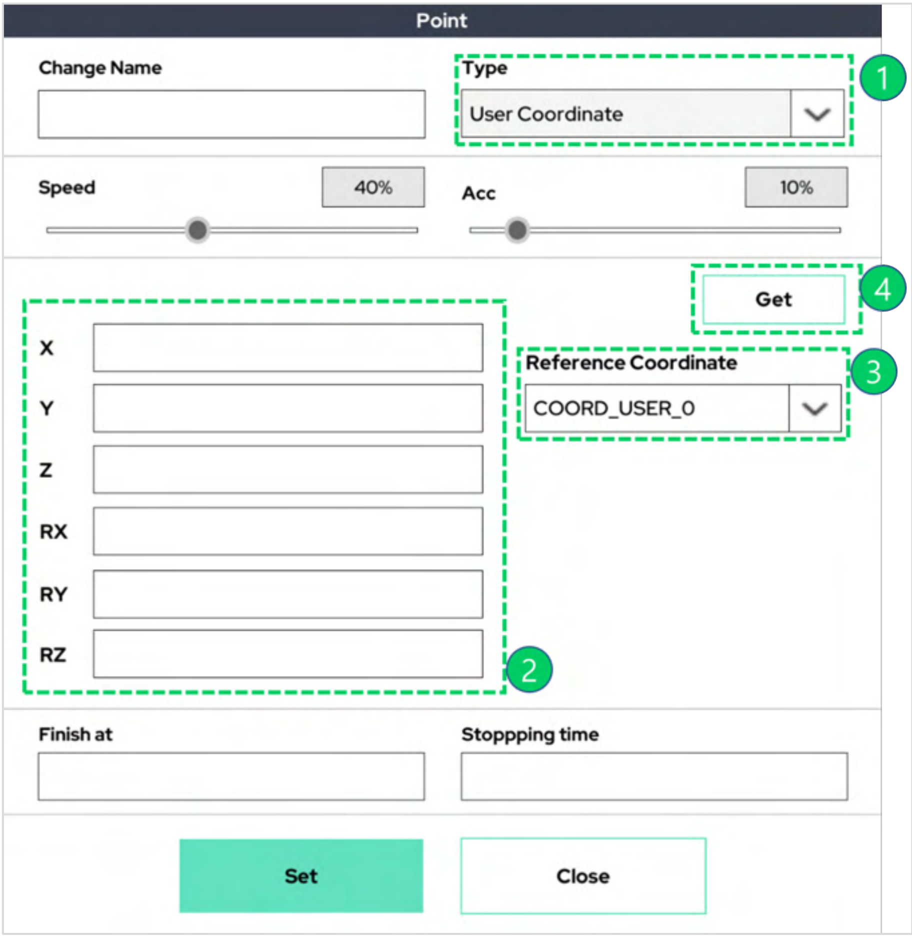
① Relative 옵션입니다.

② 상대적으로 이동할 거리/각도 offset 을 입력합니다. 변수화 된 정보들을 활용할 수도이 있습니다.

③ 상대적으로 이동할 지점을 고르는 기능입니다. 기본 값은 PT_LAST_TCP 로써 직전 도착 지점을 나타냅니다. 이외에도 임의의 point 를 기준으로 할 수도 있습니다.

④ 어떤 좌표계를 기준으로 상대이동을 지정할 것인지를 정할 수 있습니다.
기본값은 Frame_Base 로써 로봇 팔의 Base 좌표계를 나타냅니다. 유저 좌표계나, 툴의 local 좌표계로 변경 가능합니다.

▶ Linear 타입 - User coordinate 옵션



① User Coordinate(사용자 좌표계) 옵션입니다.

② Variable 과 비슷하지만, 유저 설정 좌표계를 기준으로 목표 지점을 설정하는 기능입니다. 변수화 된 정보를 입력할 수도 있습니다.

③ 기준으로 사용하고 싶은 유저 좌표계를 선택 가능합니다.

④ 원하는 기준 좌표계를 선택하고, Get(가져오기) 기능을 활용하면, 현재 로봇의 자세/위치 정보가 선택된 좌표계 기준으로 자동 입력할 수 있습니다.

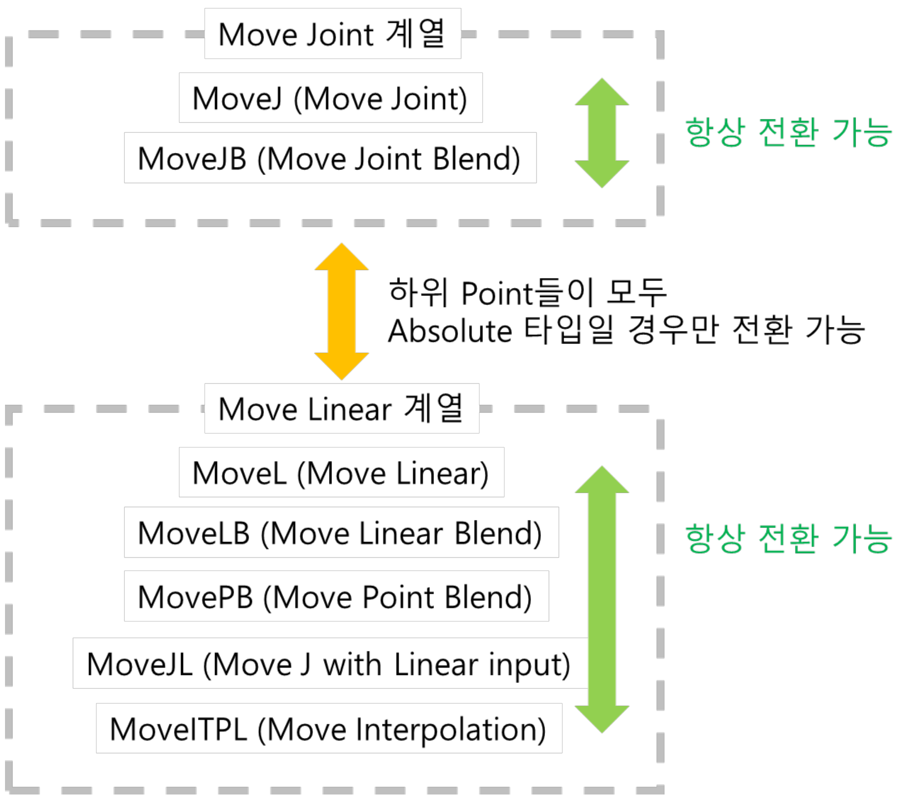
주의:

1) 유저 좌표계(User-Coordinate)에 대한 설정은 Setup 화면의 Coordinate 메뉴나, Make 화면의 Setting 기능을 통해 설정 가능합니다.
2) 최대 3 개의 유저 좌표계를 설정하고 사용 가능합니다.
3) 공장 출하 시 기본 유저 좌표계는 로봇 베이스 좌표계와 같은 좌표계로 세팅 되어 있습니다.

■ 동작 속성의 전환

이미 구성된 동작에서 동작 속성(Move 의 종류)을 바꿀 때는 다음과 같은 조건이 적용됩니다.

- 같은 계열에서의 전환은 어떠한 제한없이 전환 가능합니다.
- 다른 계열 (Move Joint 계열 -> Move Linear 계열 / Move Linear 계열->Move Joint 계열)로 전환 시에는, 하위에서 사용하고 있는 Point 기능의 타입(옵션)이 Absolute 일 경우에만 전환 가능합니다.



■ 동작 프로그램 티칭하기/만들기 예시

아래의 내용은 위의 Move 기능, Point 기능을 바탕으로 간단한 프로그램을 만들고 구동하는 예시입니다.

[Step 1]

새로운 프로젝트를 생성합니다. 프로젝트 이름은 test 로 만들었습니다.

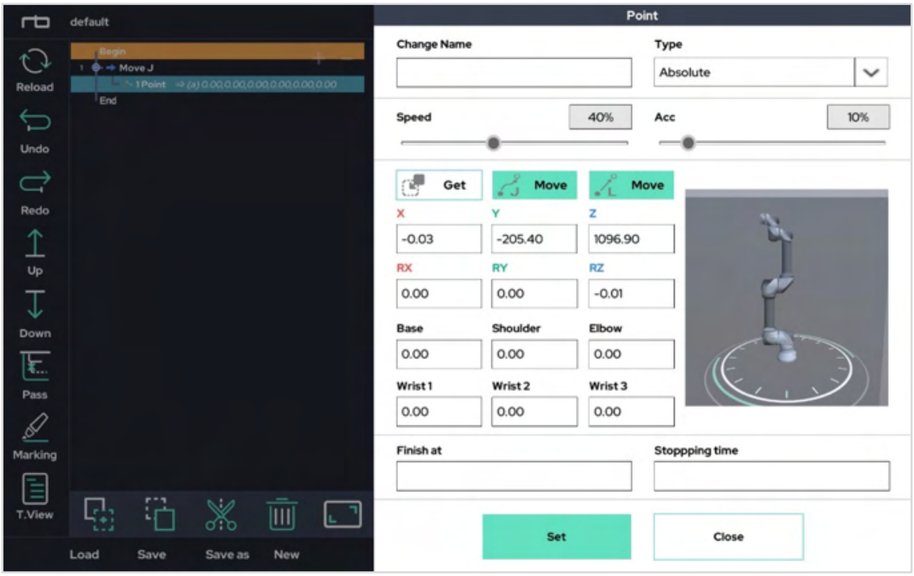
[Step 2]

프로그램 tree 에 Move 기능  추가합니다. Tree 에 MoveJ 라는 속성으로 움직이는 Point 하나가 아래와 같이 생성됩니다.

[Step 3]

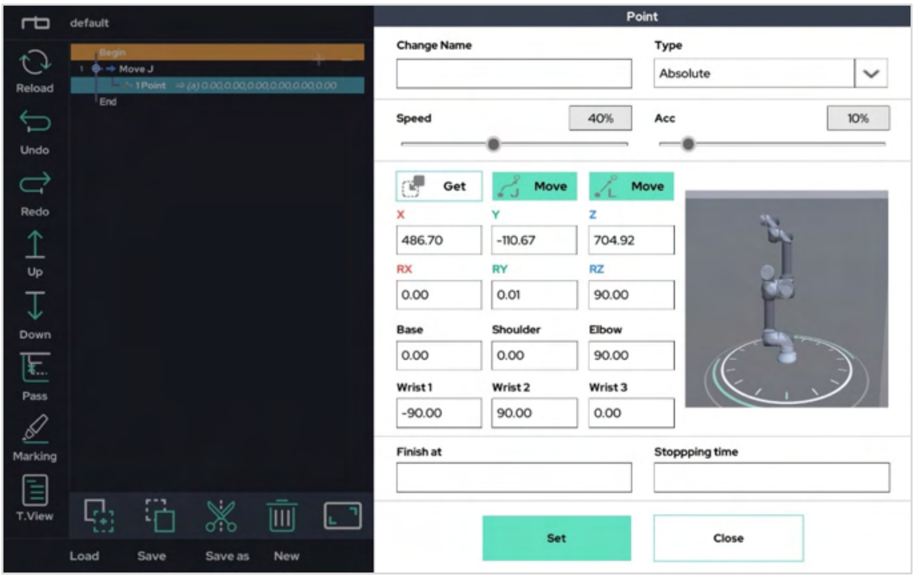
조그를 통해 로봇을 특정 자세로 이동시킵니다. 본 예제에서는 로봇의 관절 각도가 [Base:0', Shoulder:0', Elbow:90', Wrist1:-90', Wrist2:90', Wrist3:0']가 되도록 각 축을 움직입니다.

프로그램 tree 에서 Point 기능을 클릭하면 아래와 같이 Point 설정 팝업 창이 나타납니다.



[Step 4]

Point 팝업 창에서 Get(가져오기) 버튼을 눌러서 현재 로봇의 자세/각도 값을 불러옵니다. 마지막에 움직였던, 자세 각도가 가져와 집니다. Set(설정)을 눌러서 이 설정을 Point 에 저장합니다.



[Step 5]

위의 과정을 한 결과는 아래와 같습니다.

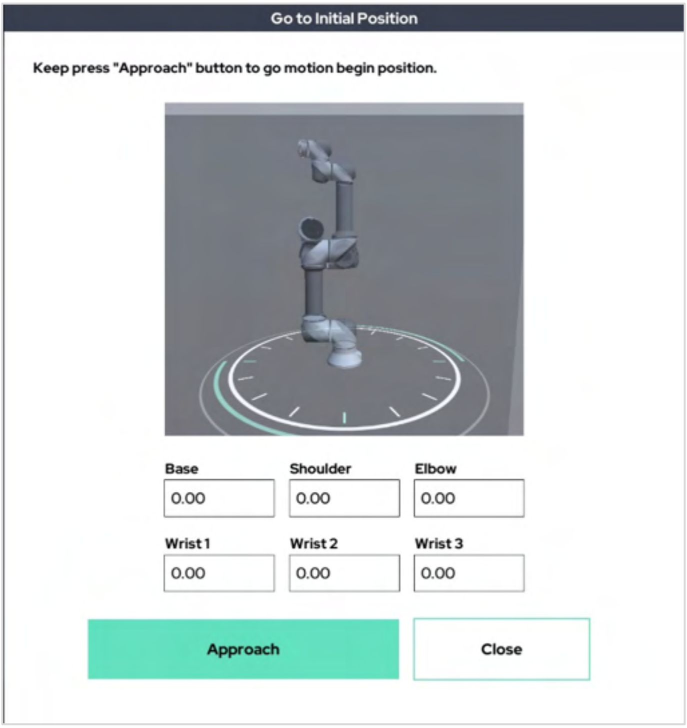
[Step 6]

위의 Step 1~4 의 과정을 몇차례 반복하여 원하는 로봇 동작을 티칭 합니다. 완성된 예시 프로그램의 구조는 아래와 같습니다.

[Step 7]

모든 티칭이 끝나고 나면 작업화면에서 티칭된 프로젝트를 실행하여 실제 로봇 팔을 움직여 볼 수 있습니다. 만약 시뮬레이션으로 동작을 해보려면 Simulation 모드를 사용하고, 실제 로봇을 구동하려면 Real 모드를 사용하면 됩니다.

작업화면 하단에 있는 플레이 버튼(▷)을 클릭하시면 아래 그림과 같이 로봇 팔을 시작 자세(begin 에서 정의된 자세)로 이동시키는 화면이 나타나게 됩니다.

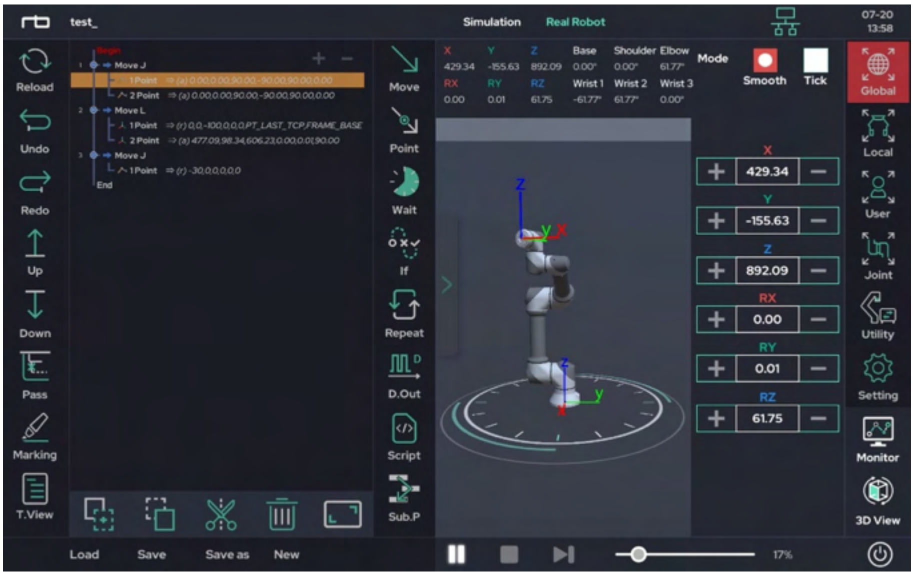


여기서 Move 버튼을 누르고 있으면 로봇 팔은 시작 자세로 이동합니다. 이동이 끝나면 시작 자세에 도달하였다는 팝업 메시지가 나타납니다.

[Step 8]

로봇이 시작 자세에 도달하였다면, 프로그램이 실행될 준비가 되었습니다. 하단의 플레이 버튼을 다시 클릭하면 프로그램이 실행됩니다.

아래는 프로그램이 실행중인 모습입니다.



주의

주의:

1) 현재 동작 중인 부분이 UI 상에 노란색으로 표기됩니다.

■ 동작 시작 자세

동작 시작 자세는 프로그램의 Begin 항목에 지정되어 있습니다.

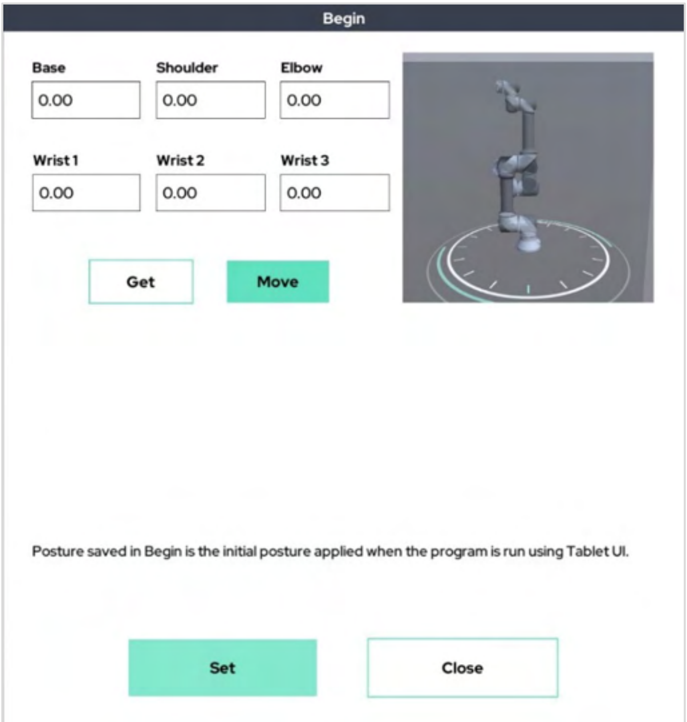
태블릿 UI 를 활용하여 로봇 구동 시, 프로그램을 최초 시작하기에 앞서 항상 Begin 자세(시작 자세)로 이동합니다.

시작 자세는 아래와 같은 방법으로 변경 가능합니다.

1) 원하는 시작 자세로 로봇을 움직입니다. 조그나 직접 교시를 사용합니다.

2) 프로그램 트리에서 Begin 부분을 클릭합니다.

3) Get(가져오기) 버튼을 클릭하여 현재 로봇 자세 각도를 입력하고 Set(설정) 버튼을 누르면 이 자세가 시작 자세가 됩니다.



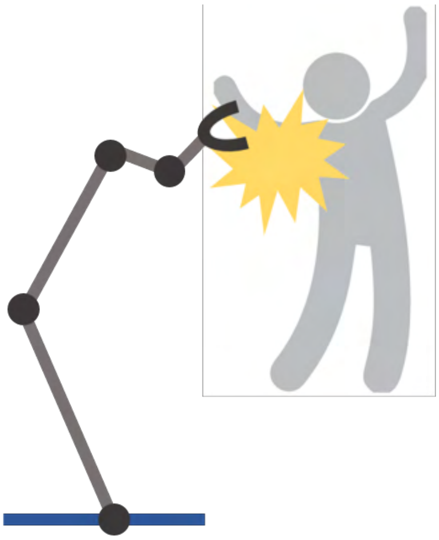
주의:

1) 프로그램 최초 생성시 기본 시작 자세(Begin 에 설정된 각도 값)은 모든 관절 각도가 0 인 자세입니다.

■ 동작 중 충돌 감지 기능

본 제품은 두 가지 충돌 감지 기능이 내장되어 있습니다.

- 외부 충돌 감지 기능 (Out-Collision Detection)
- 자가 충돌 감지 기능 (Self-Collision Detection)



[외부 충돌: External Collision]

[자가 충돌: Self Collision]

● 외부 충돌 감지 기능(Out Collision Detection)

-사전에 정의되지 않은 외부 충격/충돌을 감지합니다.

-사람 또는 외부 환경과의 예상치 못한 충돌을 감지합니다.

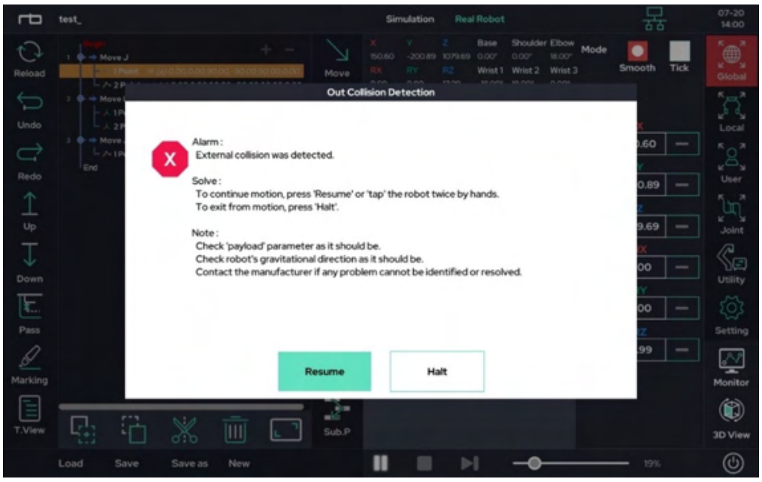
-충돌 감도는 Setup 에서 변경 가능합니다.

-프로그램 실행 중 실시간으로 충돌 감도를 Set 기능을 통해 변경 가능합니다.

-정확한 충돌 감지 기능을 위해서는 Tool 의 부하/무게 중심 등이 정확히 세팅 되어야 합니다.

-민감한 충돌 감지 설정으로 운용 시, 급격한 가속/감속을 가진 동작 등에서 충돌로 인지할 수 있습니다.

로봇 팔이 Real 모드로 실제 동작 중 외부 충돌이 감지되면 아래와 같은 팝업이 나타납니다.



- Resume(계속) 버튼: 상황을 확인하고 동작을 계속합니다.
- Halt(정지) 버튼: 프로그램을 종료합니다.

위의 팝업 창에서 로봇의 동작을 계속하려면 Resume(계속) 버튼을, 프로그램을 종료하려면 Halt(정지) 버튼을 누르면 됩니다.

이외에도 로봇을 손으로 ‘톡 톡’ 두 번 쳐주게 되면 Resume(계속) 버튼을 누른 것과 같이 동작이 계속 진행됩니다.

● 자가 충돌 감지 기능(Self-Collision Detection)

-사용자가 특정 동작 명령을 주었을 때 로봇이 로봇 자신과 스스로 충돌하는 것을 미리 예측하여 멈춥니다.

-로봇 자기 자신뿐만 아니라 미리 설정된 주변환경 (Workspace) 범주를 넘어서게 되면 스스로 멈춥니다. 주변 환경 영역에 대한 설정은 Setup 화면에서 진행합니다.

-또한 툴에 가상의 충돌 감지 박스를 설정하여 해당 박스가 로봇 자신과 충돌이 예상되거나 주변 환경 영역을 벗어나게 될 것이 예상되면 로봇이 스스로 멈춥니다. 툴에 설치된 가상의 박스는 Setup-Tool 에서 설정합니다.

아래 그림은 사용자가 조그를 오작동 시켜 로봇을 스스로 충돌하게 만든 상황입니다. 충돌직전 로봇은 스스로 충돌을 예측하여 멈추고, 사용자 UI 에 붉은색으로 경고를 나타냅니다.

아래 그림은 사용자가 로봇 구동/티칭 중 설정된 작업 영역(Workspace)를 벗어나려 한 경우입니다. 로봇은 Workspace 를 벗어나기 직전 스스로 멈추고, 사용자 UI 에 붉은색 경고를 보여줍니다.

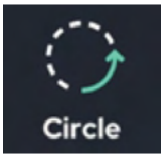
아래 그림은 사용자가 툴에 가상의 충돌 방지 영역 박스를 설정하고, 로봇을 구동 중 해당 박스 영역이 로봇과 충돌하려는 경우입니다. 로봇은 스스로 충돌을 예측하여 멈추고, 사용자 UI 에 붉은색으로 경고를 나타냅니다.

Real 모드로 운용 중 자가 충돌 감지로 멈출 시, 직접 교시 기능을 통해 안전 위치로 로봇 자세를 변경 후 계속 사용하세요.

7.4 주요 티칭 아이콘 상세설명

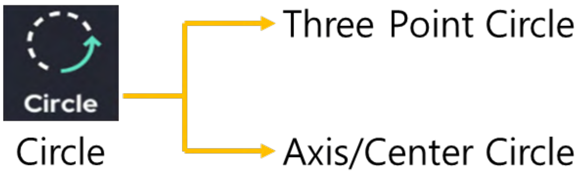
앞서 6.3 에서는 로봇 티칭의 기본이 되는 Move 기능과 Point 기능을 서술하였습니다. 본 섹션에서는 기타 티칭 아이콘 / 기능 아이콘들에 대한 설명입니다.

■ Circle 기능 :



Move 와는 별도로 원형 동작을 위한 기능을 제공합니다.

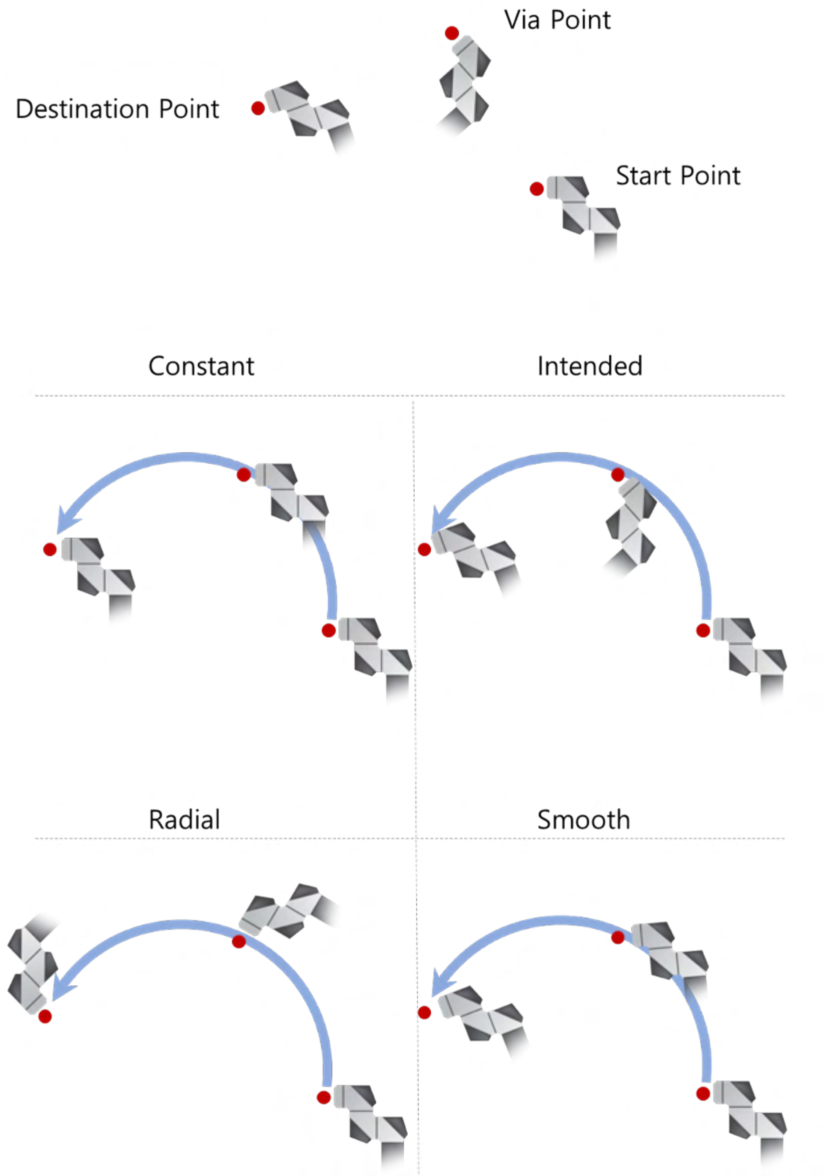
시작점과 두 점을 찍어서, 3 점을 지나는 호(arc)를 그리는 타입이 존재하고, 중심점과 회전축을 설정하여 원을 그릴 수 있는 타입이 있습니다.



2) Three Point Circle: 3 점의 정보를 가지고 호를 그리는 모드

3) Axis/Center Circle: 회전 중심과 회전 축 정보를 가지고 호/원을 그리는 모드

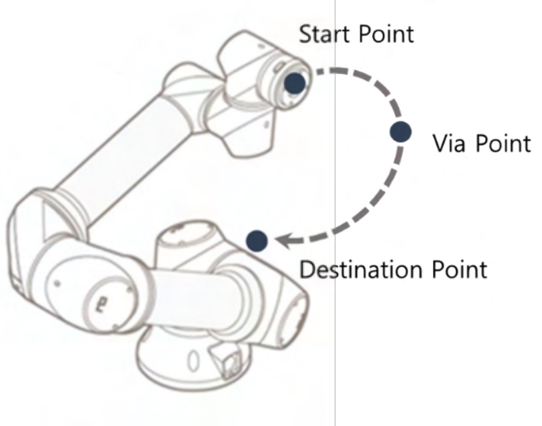
원을 그리는 타입과 별도로, Circle 기능에서는 아래와 같이 4 가지 회전 옵션(Orientation Option)이 제공됩니다.

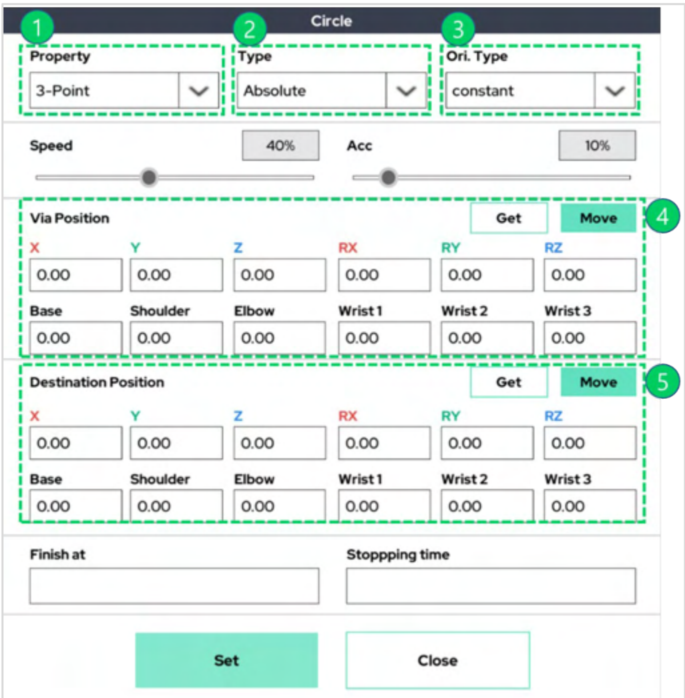


- Constant: 회전 동작을 하면서 말단(TCP)의 회전 자세를 유지합니다.
- Intended: 유저가 저장한 회전 자세를 그대로 따라갑니다.
- Radial: 회전 동작의 중심점을 기준으로 말단(TCP)의 회전 자세를 함께 회전시킵니다.
- Smooth: 시작점에서 도착지점으로 회전이 바로 바뀝니다. 경유지점의 회전 정보는 무시됩니다.

Three Point Circle(3 점 설정 타입)

시작 점과, 중간 경유점, 그리고 도착점, 3 점의 정보를 바탕으로, 3 점을 연결하는 호 (arc)를 그리는 타입입니다.





① 원형 동작을 위한 타입(3 점 설정 타입)

② 점의 속성 (Absolute / Variable / Relative / UserCoord.)

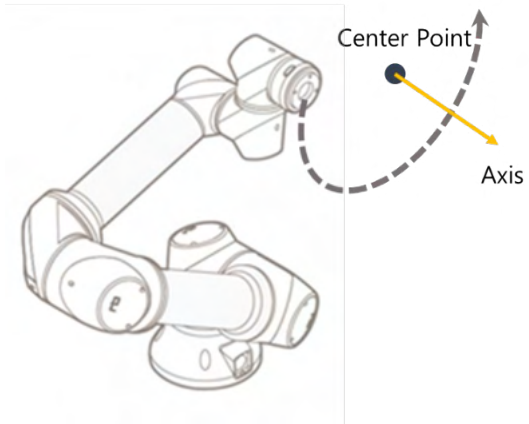
③ 회전 옵션 (Constant / Radial / Intended / Smooth)

④ 경유점(Via Point)에 대한 정보

⑤ 목표점(Destination Point)에 대한 정보

Axis/Center Circle(축/중심 설정 타입)

원형 동작을 위한 중심점과 회전 축, 그리고 회전할 각도를 설정하면 원형 동작을 하는 타입입니다.



① 원형 동작을 위한 타입(축/중심 설정 타입)

② 점의 속성 (Absolute / Variable / Relative / UserCoord)

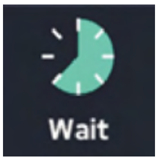
③ 회전 옵션 (Constant / Radial / Intended / Smooth)

④ 중심점(Center Point)에 대한 정보

⑤ 회전축(Axis)에 대한 정보

⑥ 회전 각도(degree)에 대한 정보

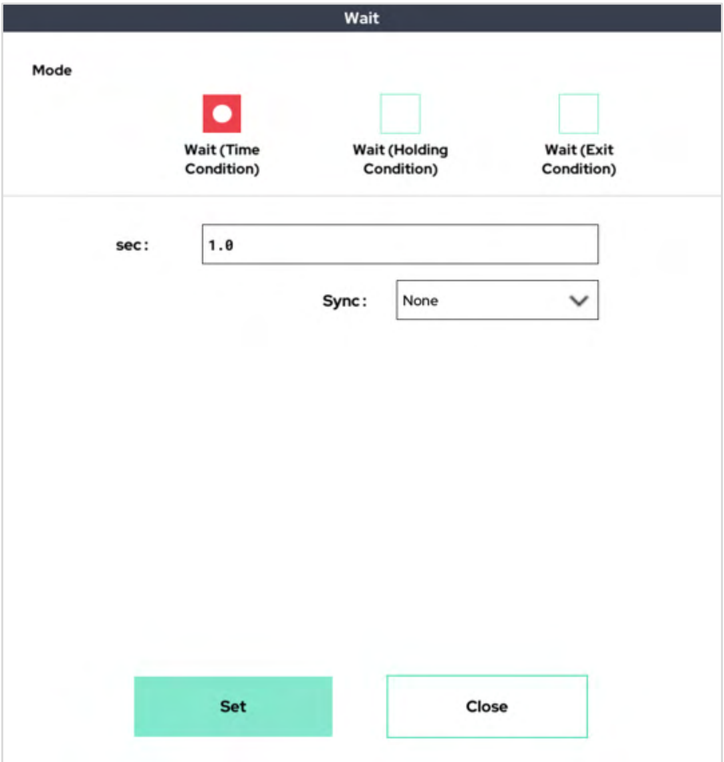
■ Wait 기능:



지정된 조건이나 시간만큼 기다리는 기능입니다.

3 가지 모드가 존재합니다. 지정된 시간만큼 기다리는 기능, 조건이 참일 동안 기다리는 기능, 그리고 조건이 참이면 기다림을 종료하는 기능이 있습니다.

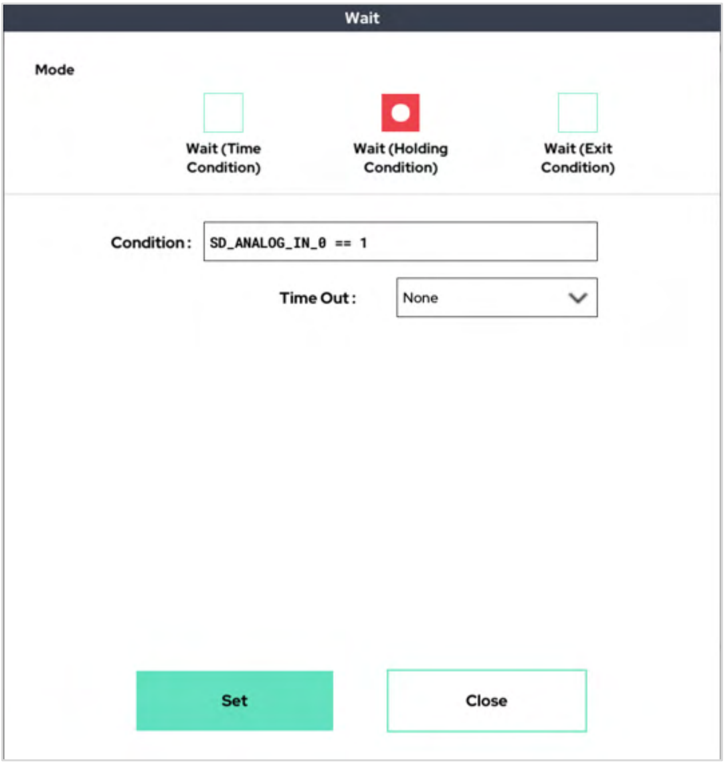
1) 지정된 시간만큼 기다리는 기능(Time Condition)



예) 지정된 시간(3 초) 만큼 기다리고 다음 명령어 실행

Sync 에서 속도 조절바 동기화 기능을 사용할 경우, 속도 조절바 값에 반비례하게 대기 시간이 조절된다.

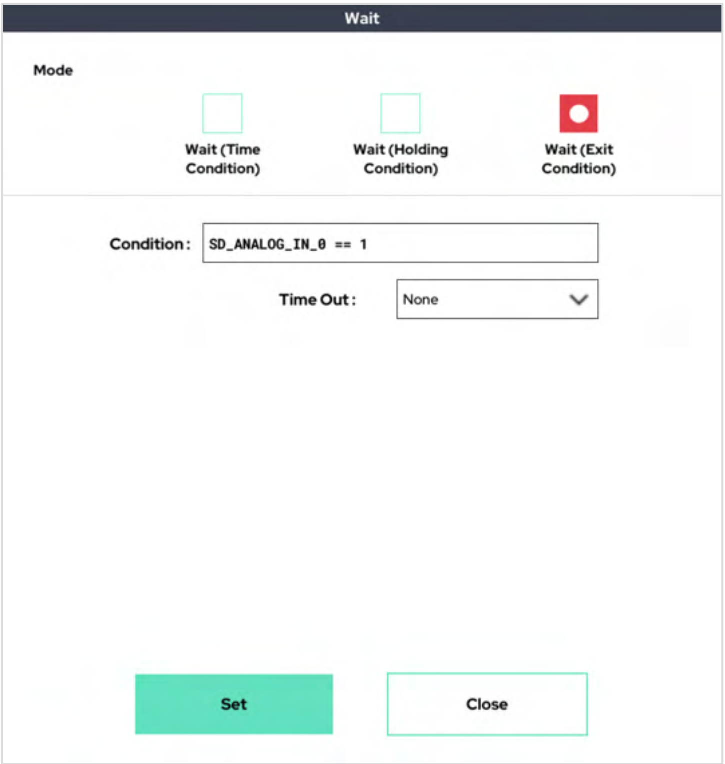
2) 지정된 조건이 참일 동안 기다리는 기능(Holding Condition)



예) Condition 에 입력한 조건문이 참이면 계속 기다림.

Time Out 기능은 조건문이 거짓이 될 수 없는 상황에서 거짓이 될 때까지 계속 기다리는 것을 방지하기 위한 기능이다. 작성한 시간이 지나면 대기를 탈출한다.

3) 지정된 조건이 참이면 기다림을 종료하는 기능(Exit Condition)



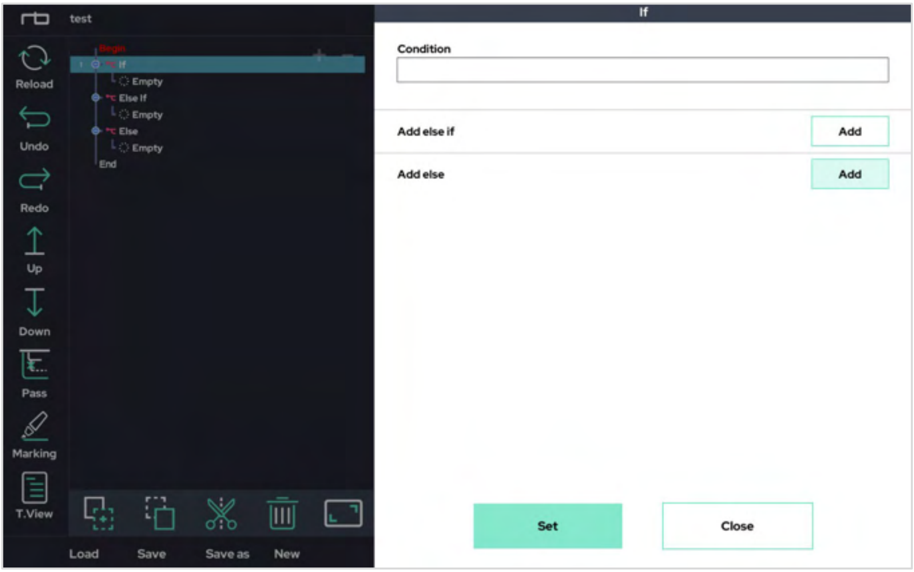
예) Condition 에 입력한 조건문이 참일 때 기다림을 멈추고 다음 명령어 수행.

Time Out 기능은 조건문이 참이 될 수 없는 상황에서 참이 될 때까지 계속 기다리는 것을 방지하기 위한 기능이다. 작성한 시간이 지나면 대기를 탈출한다.

■ If 기능:

If 문입니다. 조건에 따라 로봇이 다른 명령을 수행할 수 있도록 분기를 설정할 수 있습니다. If/else if /else 문을 사용할 수 있습니다.

If 기능을 프로그램 Tree 에 추가 후, 추가된 If 기능을 클릭해보면 아래와 같은 팝업 창이 뜹니다. 여기서 If 문에서 사용할 조건문을 직접 입력합니다.

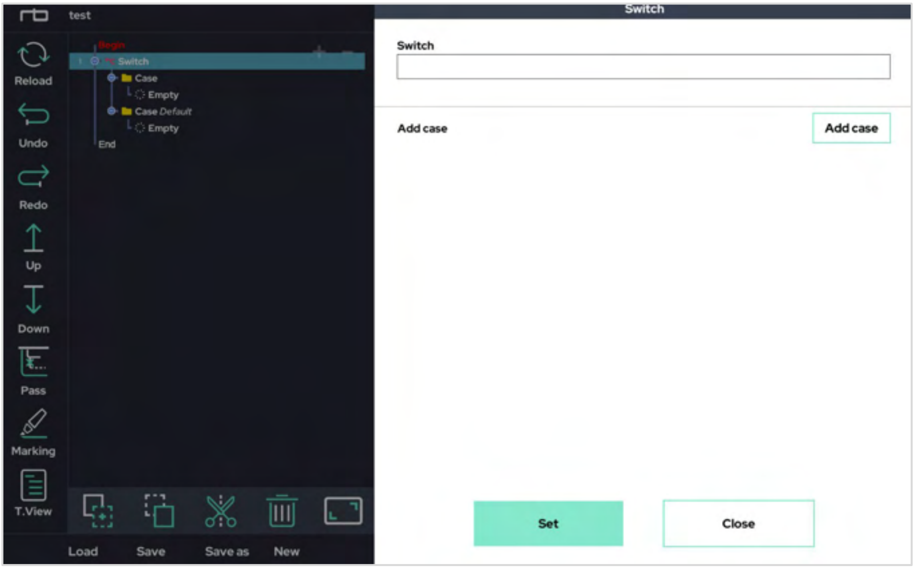


Else if 또는 Else 기능이 필요한 경우(Add else if) 버튼이나(Add else) 버튼을 눌러서 조건문의 분기를 만들 수 있습니다.

■ Switch 기능:

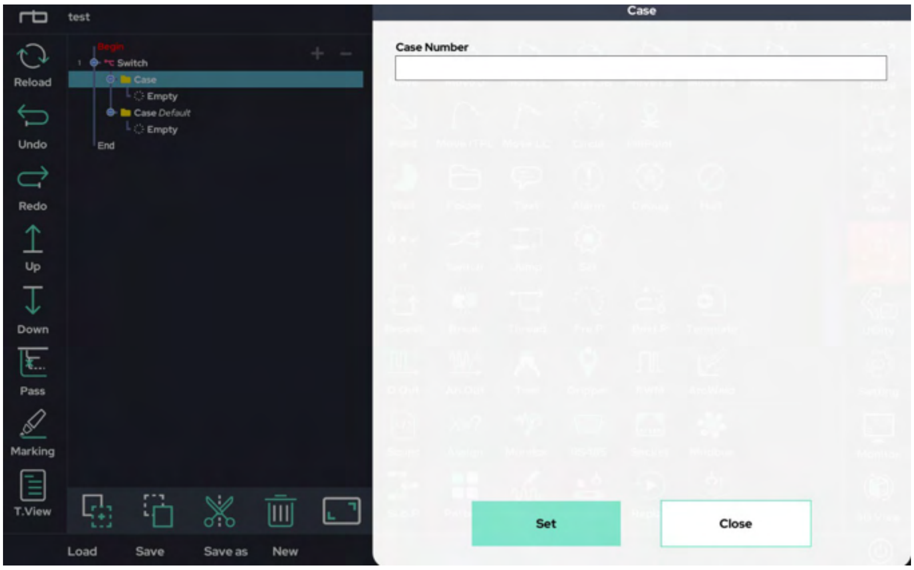
Switch 문입니다. 조건에 따라 로봇이 다른 명령을 수행할 수 있도록 분기를 설정할 수 있습니다. Switch/case 문을 이용할 수 있습니다.

Switch 기능을 프로그램 Tree 에 추가 후, 추가된 Switch 기능을 클릭해보면 아래와 같은 팝업 창이 뜹니다. 여기서 Switch 문이 동작할 기준 인자를 입력합니다.

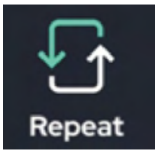


최초로 Switch 문 생성시 default 는 자동으로 생성됩니다. Case 문이 추가로 필요한 경우(Add case) 버튼을 이용하여 Case 문을 추가할 수 있습니다.

추가된 Case 문을 클릭해 보면 아래와 같은 창이 뜨게 되고, Switch 문에 넣은 기준 인자를 판별하여 들어갈 Case 문 인자를 이 창에 입력하면 됩니다.


test
Reload
Undo
Redo
Up
Down
Pass
Marking
T.View
Switch
Case
Empty
Case Default
Empty
End
Load
Save
Save as
New
Case
Case Number
Set
Close

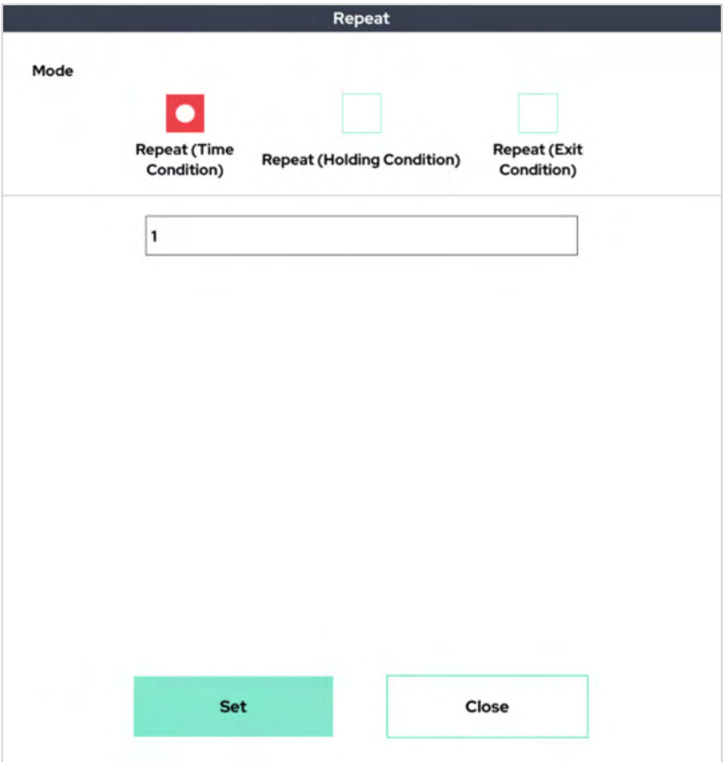
■ Repeat 기능:



반복문입니다. 지정된 조건이나 횟수만큼 하위의 프로그램을 반복합니다. 3 가지 모드가 존재합니다. 지정된 횟수만큼 반복 기능, 지정된 조건이 참일 동안 반복하는 기능, 그리고 조건이 참이면 반복을 끝내는 기능이 있습니다.

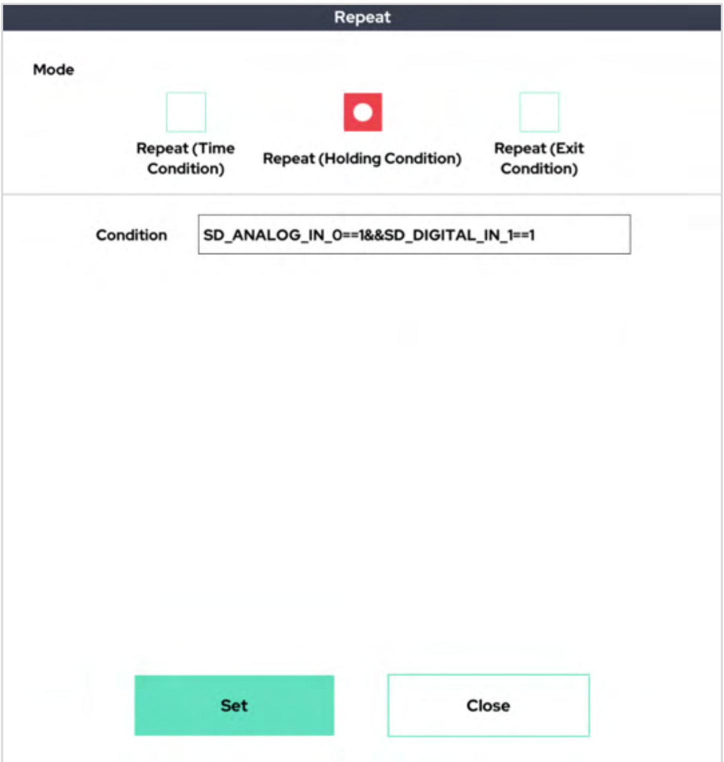
Repeat 기능을 프로그램에 추가하고, Repeat 을 클릭해보면 아래와 같이 3 가지 옵션이 존재합니다. 원하는 기능을 선택하여 사용합니다.

1) 지정된 횟수만큼 반복하는 기능(Time Condition)



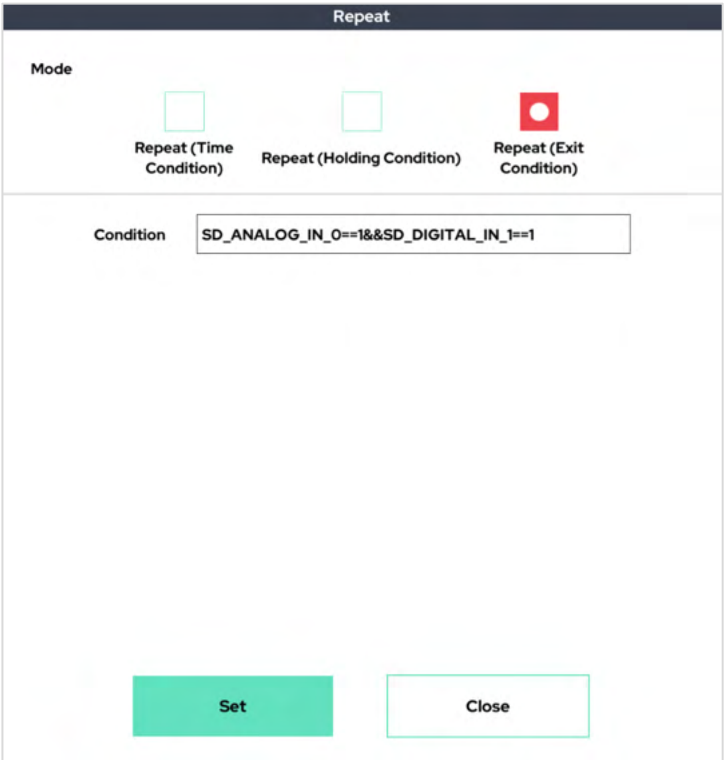
예) 1 번 반복합니다.

2) 지정된 조건이 참일 동안 반복하는 기능(Holding Condition)



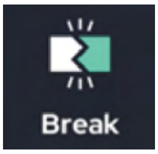
예) Condition 에 입력한 조건문이 참이면 계속 반복합니다.

2) 지정된 조건이 참이면 반복을 끝내는 기능(Exit Condition)



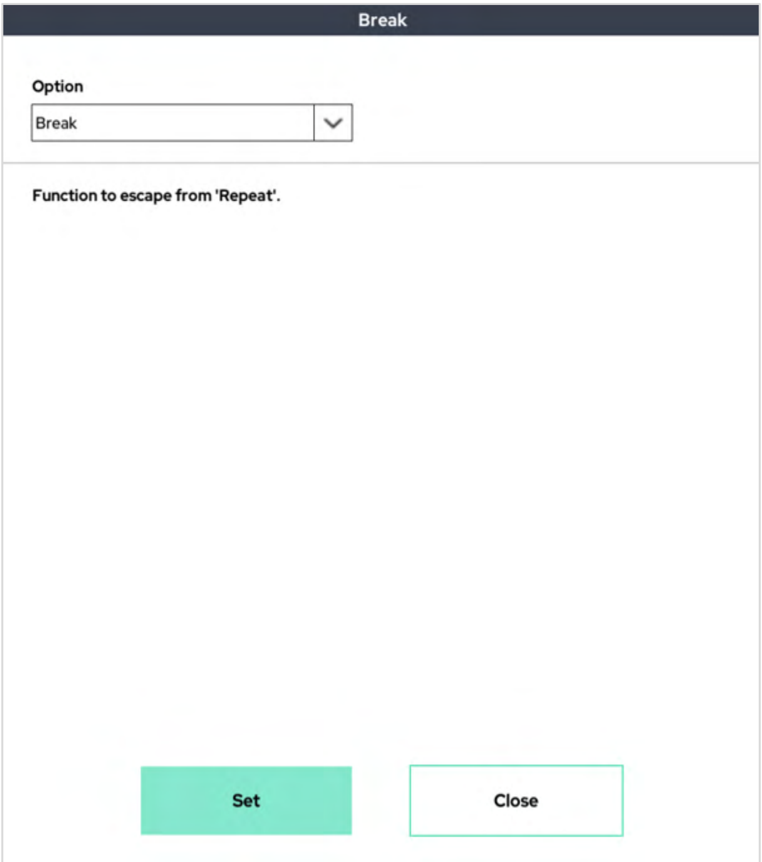
예) Condition 에 입력한 조건문이 참일 때 반복을 멈추고 다음 명령어를 수행합니다.

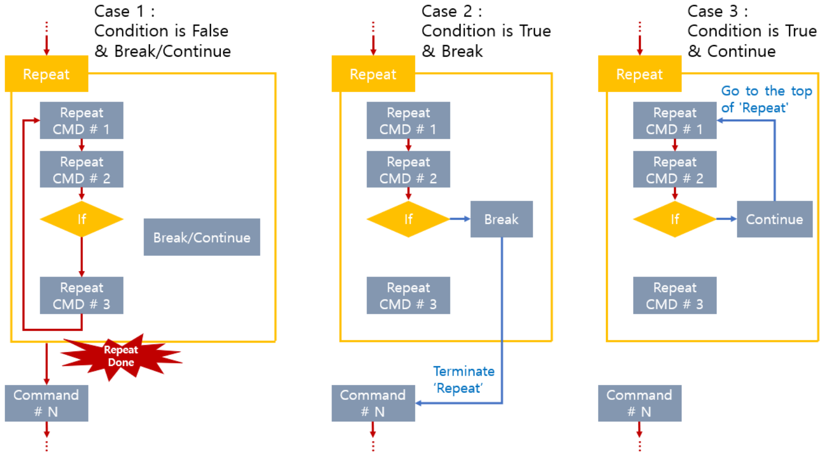
■ Break 기능:



반복문을 강제로 종료(반복 중단) 또는 반복문의 상단으로 이동(계속)하는 기능입니다.
반복문(repeat) 기능 하위에서 사용되며, break 기능은 반복문에 특정 조건이 선언되어 있더라도 사용되면 선행적으로 반복문을 탈출합니다. Continue 기능 또한 반복문 기능 하위에서 사용되며, 사용되면 하위 기능을 실행하지 않고 반복문의 상단으로 이동합니다.

반복문(Repeat) 기능의 하위에서만 작동합니다. 다른 구문에서는 프로그램에 영향을 주지 않습니다.

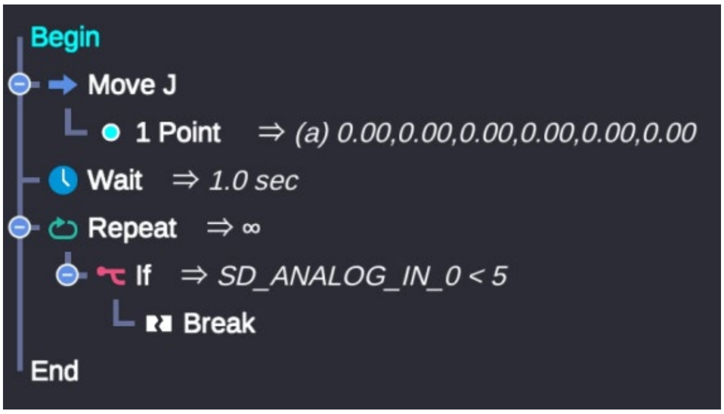




반복 중단(break)와 계속(continue)를 사용할 경우 위 그림과 같이 동작하게 된다.

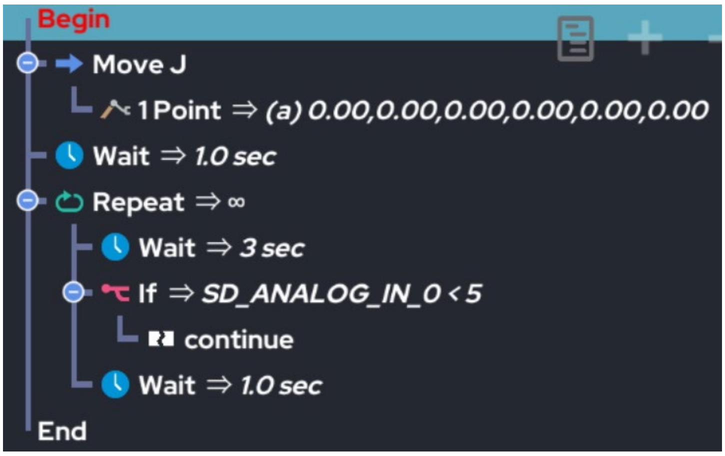
* 반복 중단(break) 예시

아래의 예시에서는 무한대의 반복문이 있습니다. 그리고 하위에 If 조건에 따라 Break 기능이 실행되도록 프로그램을 구성하였습니다. 조건이 발생하지 않는다면 반복문은 계속해서 반복하게 됩니다. 하지만 조건이 발생하는 순간 반복문을 탈출하고 다음 명령어를 실행합니다.



* 계속(continue) 예시

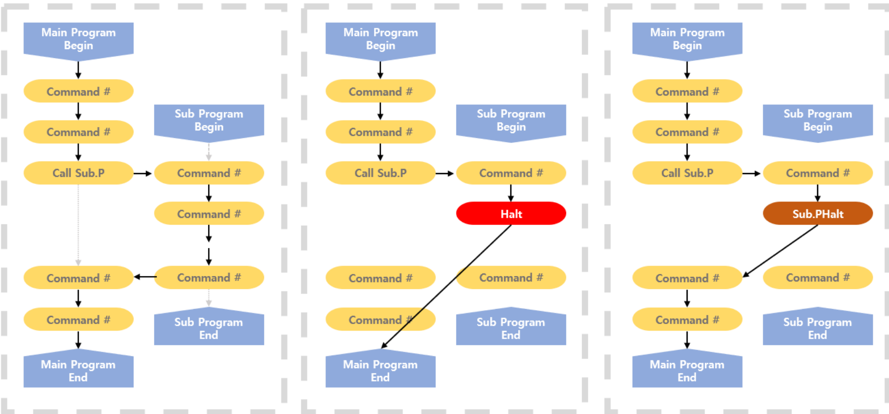
아래의 예시에는 무한대의 반복문이 있습니다. 그리고 하위에 If 조건에 따라 Continue 기능이 실행되도록 프로그램을 구성하였습니다. 조건이 발생하지 않는다면 반복문은 전체 내용을 반복하게 됩니다. 하지만 조건이 발생하는 순간 Continue 기능 하단의 내용은 실행하지 않고 Continue 기능 상단의 명령어를 실행합니다.



■ Halt 기능:

프로그램을 종료하는 Halt 기능입니다. 특정 조건이 발생했을 때 프로그램을 종료해야 한다면 이 기능을 사용하면 됩니다.

Halt 는 Halt 와 Sub.P Halt 로 나뉘게 됩니다. Halt 는 메인 프로그램, 서브 프로그램 어디에서 실행되든 메인 프로그램을 종료하는 기능입니다. Sub.P Halt 는 서브 프로그램 내에서 사용해야 하며 실행되는 순간 서브 프로그램을 종료하고 메인 프로그램으로 돌아가게 됩니다. 아래의 다이어그램을 참고 바랍니다.

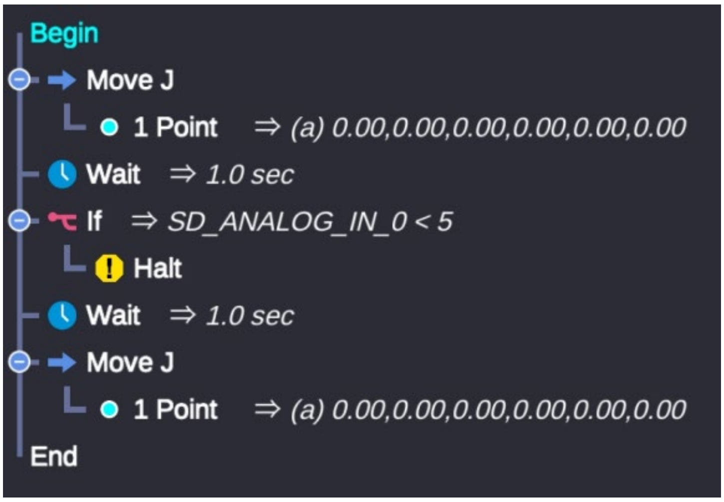


common case

during the subprogram
When Halt is called

during the subprogram
When Sub.P Halt is called

아래의 예시에서는 If 조건문을 검사하여 해당 조건문이 참일 경우 Halt 기능을 호출하도록 프로그램을 구성하였습니다. 해당 조건 문이 참일 경우, 프로그램은 해당 위치에서 종료되고, 다음 명령어들을 수행하지 않습니다.



주의

주의:

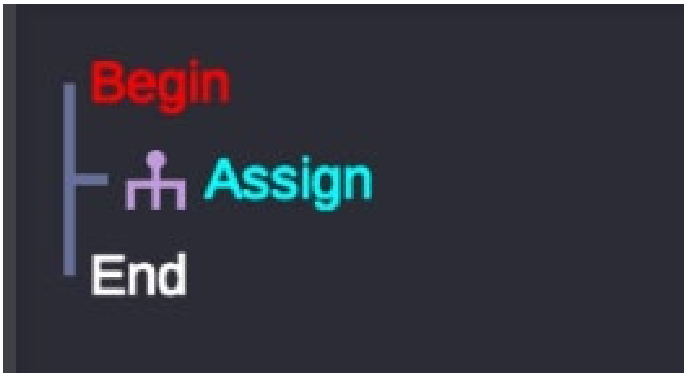
1) Halt 기능이 수행되면, 메인 프로그램뿐만 아니라 Thread(쓰레드) 기능도 함께 종료됩니다.

■ Assign 기능:

변수화 기능을 사용하기 위해, 초기 변수 선언을 위한 Assign 기능입니다. 변수 형태는 4 타입으로 선언할 수 있습니다. 단일 숫자를 저장하는 variable 타입, 배열 숫자를 저장하는 array 타입, 자세 정보를 저장하는 point 타입, 그리고 문자열을 저장하는 string 타입이 있습니다.

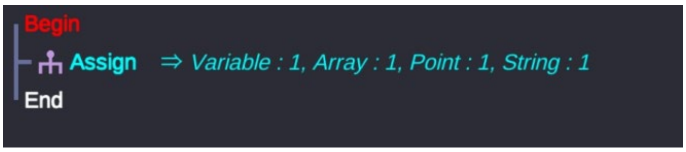
- Variable 타입: 단일 숫자 형태입니다.
- Array 타입: 배열을 저장합니다(배열의 최대 길이는 10 개입니다).
- Point 타입: 자세 정보를 저장합니다(x, y, z, Rx, Ry, Rz 를 저장합니다).
- String 타입: 문자열을 저장합니다.

Assign 기능을 프로그램 Tree 에 추가하면 아래와 같이 추가됩니다.



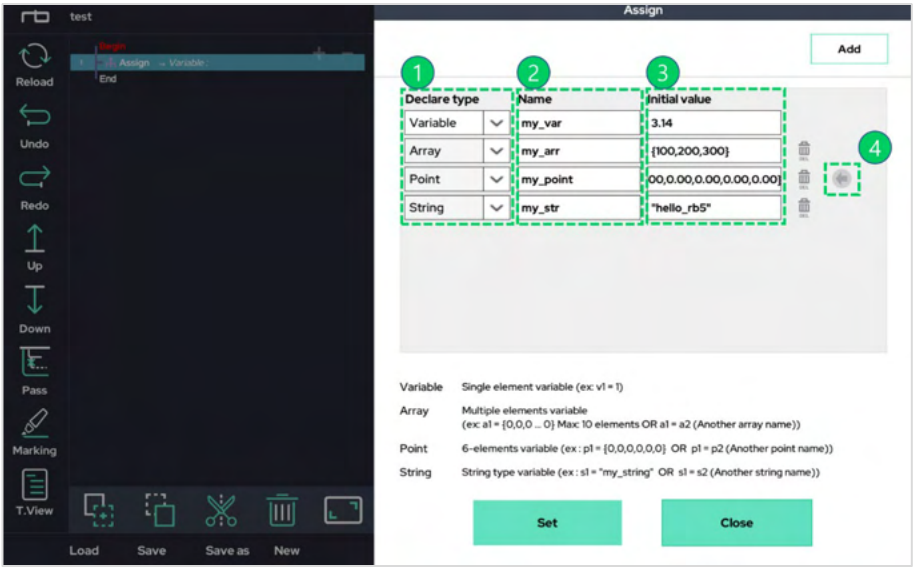
추가된 Assign 을 클릭하면 아래와 같은 팝업이 뜹니다. 여기서 선언하고 싶은 변수를 선언할 수 있습니다. 한 번에 여러 변수를 선언하고 싶을 경우(Add) 버튼을 클릭하여 한 번에 여러 개 / 여러 타입의 변수를 선언할 수 있습니다.

변수 선언이 끝나면 아래와 같이 Tree 상에 어떤 형태의 변수를 몇 개씩 선언했는지 나타납니다.



만약 선언을 하나만 하면 Tree 상에 아래와 같이 변수 이름과 초기값이 같이 뜹니다.

변수 선언 (Assign) 기능의 팝업 창 구성은 아래와 같습니다.



2) 변수의 형태를 선택합니다(Variable, Array, Point, String 중 택 1.
3) 변수의 이름을 정합니다.
4) 선언 시 초기값을 설정하는 부분입니다.

Variable 형태의 경우 단일 숫자로 초기값을 넣습니다(예: 1).

Array 형태의 경우 배열 형태로, 중괄호를 사용하여 초기값을 넣습니다(예: {100, 200, 300}).

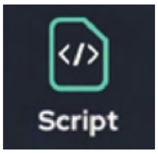
Point 형태의 경우 중괄호를 사용하여 6 개 길이의 배열 형태로 초기값을 넣습니다(예: {300, 300, 300, 0, 90, 0}).

String 형태의 경우 따옴표를 사용하여 문자열을 초기값으로 넣습니다(예: “hello_rb5”).

5) Point 형태에만 있는 버튼입니다.

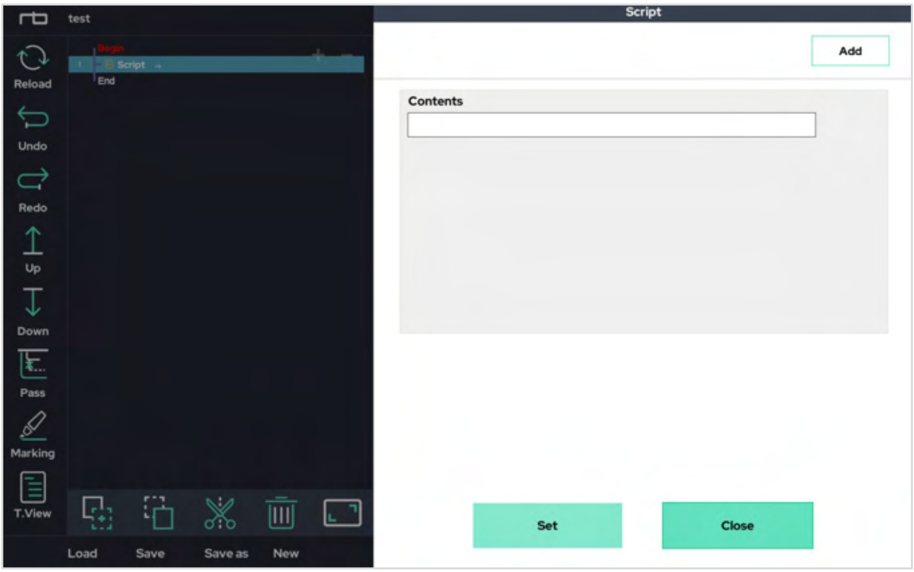
현재 로봇의 6 개 자세 값(x,y,z,Rx,Ry,Rz)를 자동으로 초기값에 넣어줍니다.

■ Script 기능:



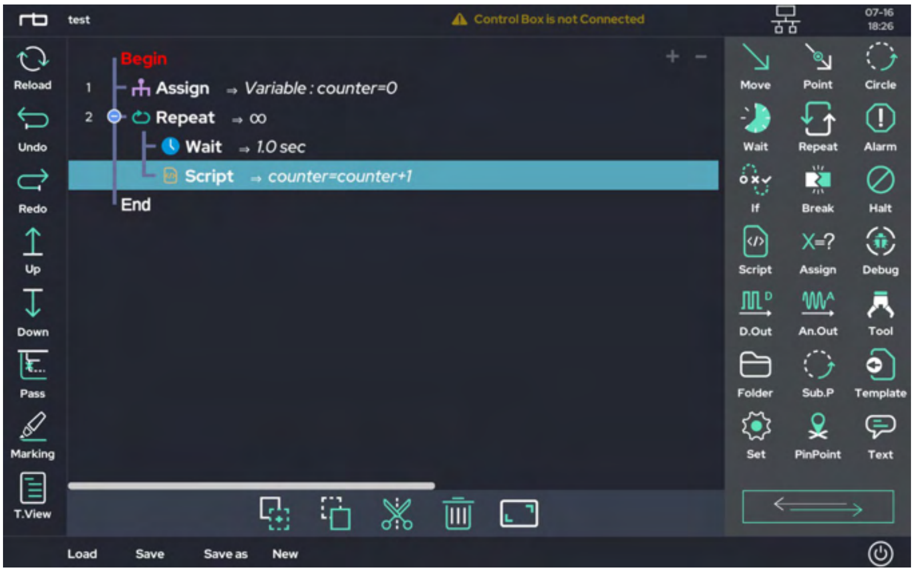
사용자가 임의의 스크립트를 작성할 수 있는 Script 기능입니다. 기능과 별도로 특정 연산/계산이 필요하거나, 변수 치환, 대입 등의 기능을 Script 기능을 사용하여 구현 가능합니다.

Script 기능을 프로그램 Tree 에 추가하고, 추가된 Script 기능을 클릭하면 아래와 같은 팝업창이 나타납니다.



이곳에 사용자가 원하는 스크립트를 자유롭게 입력 가능합니다. 변수 연산, 변수 치환, 변수 대입 등 다양한 기능을 키보드를 이용하여 자유롭게 입력 가능합니다. 동시에 여러 줄의 Script 기능을 사용하기 원하는 경우 팝업 창 하단의(Add) 버튼을 이용하여 사용할 수 있습니다.

아래의 예시는 1 초마다 반복하는 반복문(Repeat)을 생성하고, 반복할 때 마다 Assign 기능을 통해 선언한 ***counter*** 라는 변수를 Script 기능을 활용하여 1 씩 증가시키는 프로그램 예시입니다.

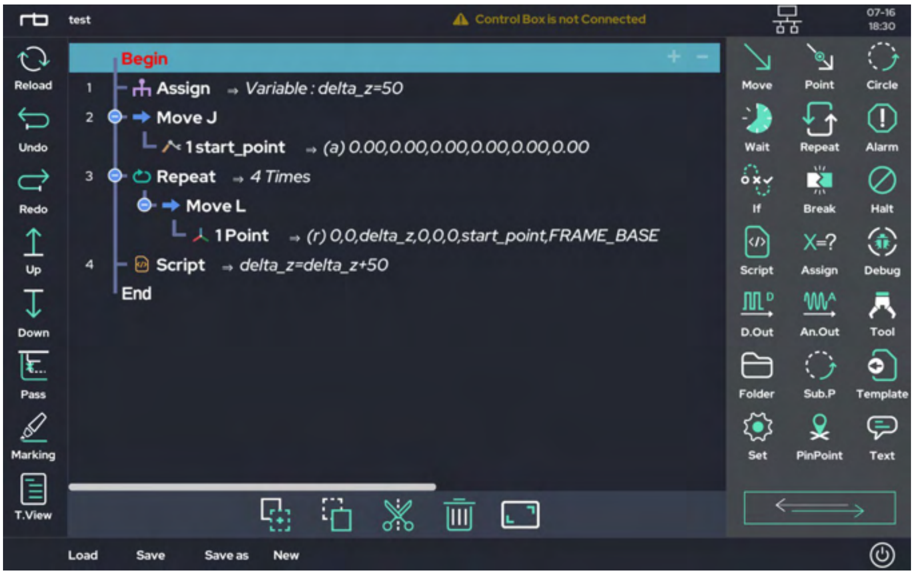


아래의 예시는 MoveJ 기능을 통해 특정 자세로 이동하고, 해당 자세의 이름을 start_point 라고 지정하였습니다(Point 의 이름 정하기 기능 사용).

그리고 반복문 (Repeat 기능)을 사용하여 MoveL 로 동작을 4 번하는 기능입니다.

- 이때 MoveL 은 Relative Point 기능을 활용하여, 앞서 선언된 start_point 대비 delta_z 만큼 z 방향으로 이동합니다. (Point 기능의 Linear movement 계열의 relative point 기능을 참고하세요)
- 이때 반복문의 마지막에는 Script 기능을 활용하여 delta_z 를 50 씩 증가시키도록 하였습니다. (delta_z 는 프로그램 상당에서 Assign 기능을 통해 미리 선언해 두었습니다.)

결론적으로 로봇은 MoveJ 로 처음 위치로 이동 후, 해당 위치를 start_point 라고 저장한 이후, 반복문에서 4 번 아래로 50mm 씩 MoveL 로 움직입니다.

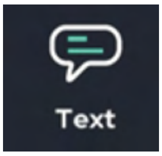


주의

주의:

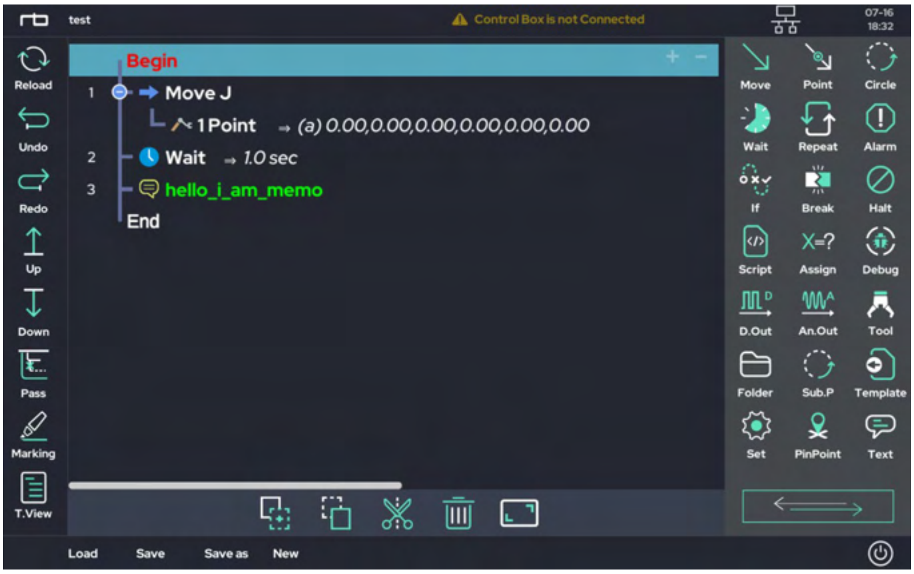
1) 스크립트 기능은 사용자가 자유롭게 스크립트를 작성하여 실행하는 영역입니다.
2) 만약 문법에 맞지 않는 스크립트 작성시, 프로그램이 오작동하거나 멈출 수 있습니다. 본 기능을 사용할 때 알맞은 문법을 사용하였는지 주의하세요.

■ Text 기능:

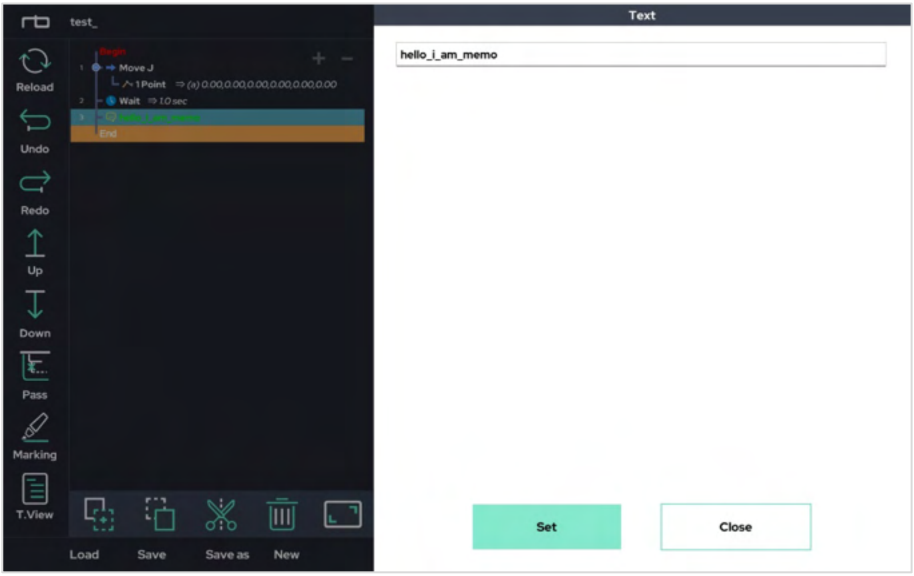


프로그램 리스트 트리에 주석과 같이 필요한 부분에 간단한 메모를 할 수 있습니다.

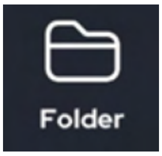
Text 기능은 프로그램 Tree 안에서 초록색으로 표기되며, 프로그램의 기능에 어떠한 영향을 주지 않습니다. 프로그램 Tree 에 Text 아이콘을 클릭하여 Text 기능을 삽입하면 아래와 같이 추가됩니다.



여기서 Text 를 클릭하여 원하는 메모를 남길 수 있습니다.

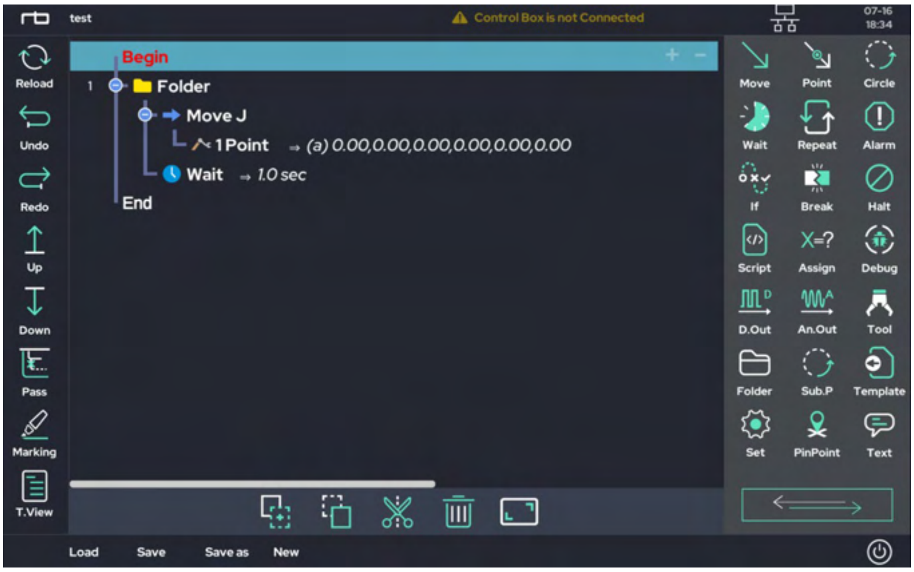


■ Folder 기능:

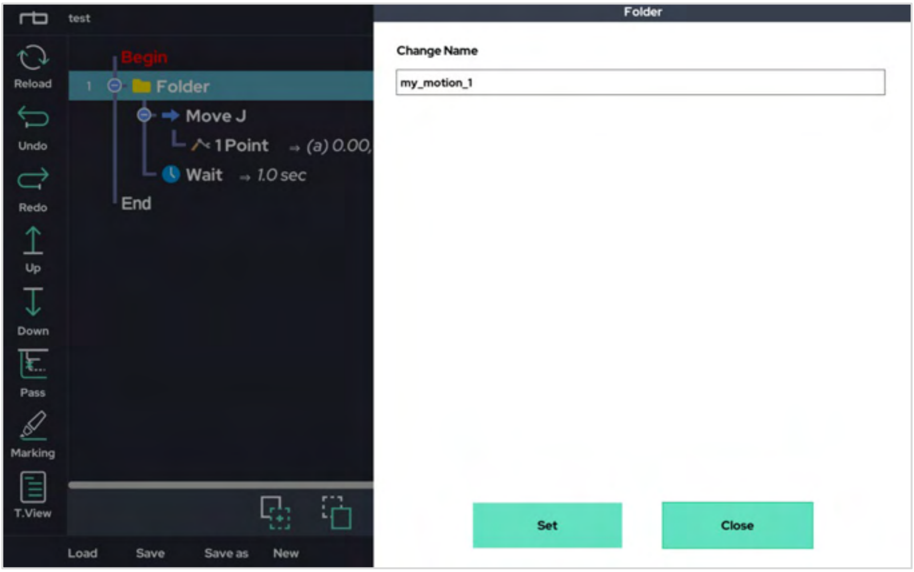


작성된 프로그램 명령들을 폴더의 하위항목으로 묶어서 모듈로 관리할 수 있도록 도와주는 폴더 기능입니다. 폴더 기능 마다 이름을 다르게 부여하여 관리를 도와주는 도구입니다.

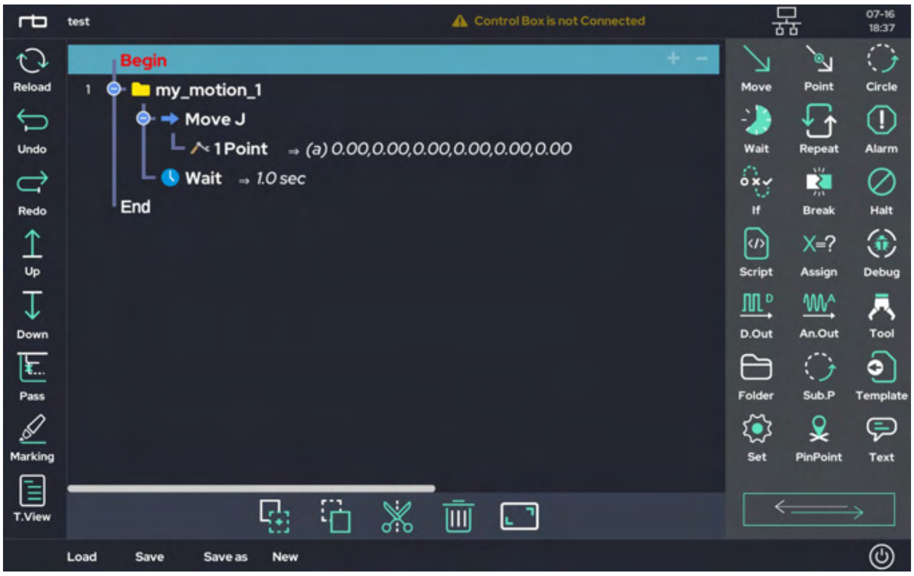
프로그램 Tree 에 Folder 기능을 추가하고, 하위 항목에 다양한 기능을 아래와 같이 넣을 수 있습니다.



여기서 Folder 의 이름을 변경하고 싶다면 Folder 아이콘을 클릭하여 이름을 변경합니다.



이름 변경 후, Tree 상에서 변경된 이름의 폴더를 볼 수 있습니다.

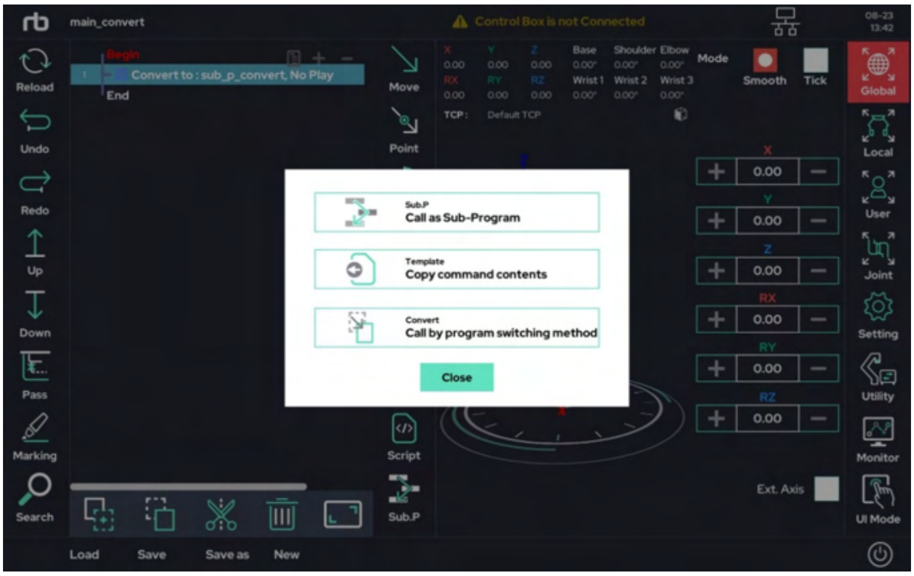


Text 기능과 같이, 프로그램의 기능에는 영향을 미치지 않습니다. 티칭 시 기능의 모듈 별 관리, 묶음 관리 등을 도와주는 기능입니다.

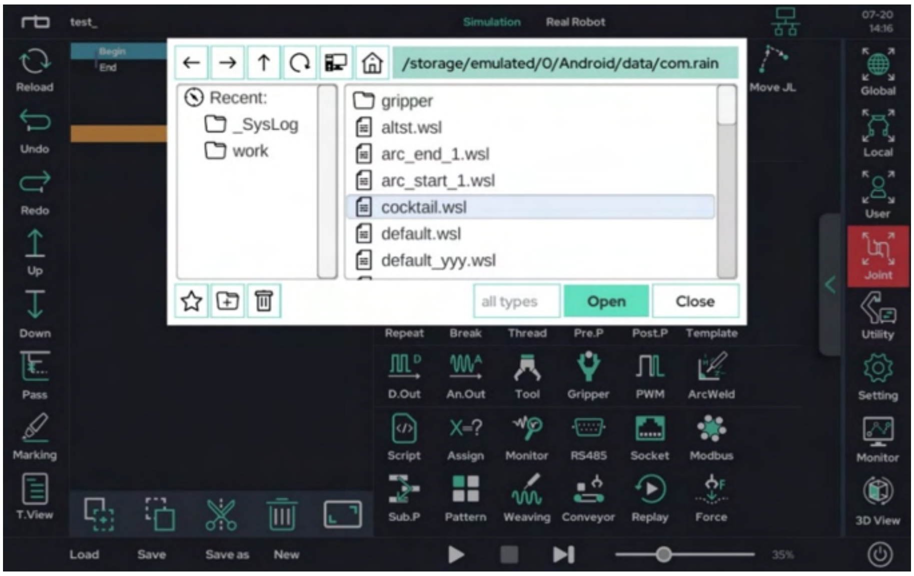
■ Sub.P(Sub Program) 기능:

사전에 만들어진 다른 프로그램 파일(티칭 파일)을 현재 프로젝트안에 삽입할 수 있는 기능입니다 (서브루틴 기능).

프로그램에서 Sub.P 아이콘을 클릭하면 아래와 같은 팝업창이 나오고 이때, 하위 프로그램으로 호출을 누릅니다.

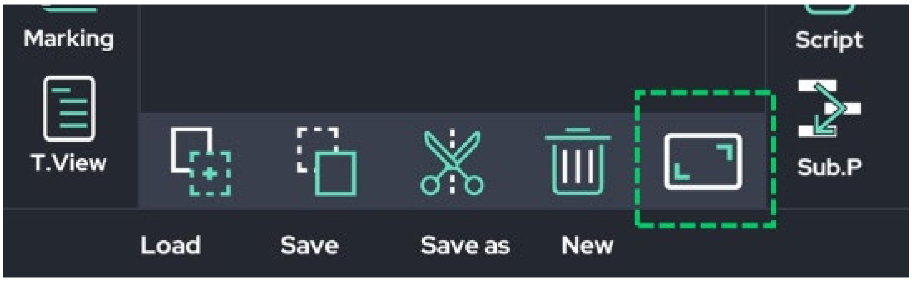


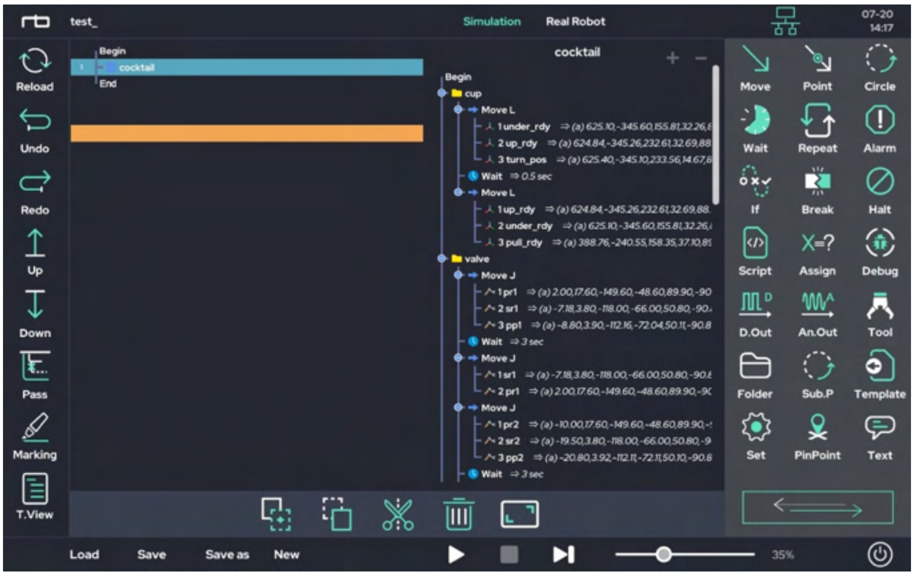
하위 프로그램으로 호출을 클릭하면 아래와 같이 파일 탐색기 기능을 갖춘 팝업창이 뜹니다.



이곳에는 해당 태블릿 PC 에서 만들어진 다른 프로젝트들이 보입니다. 여기서 원하는 프로젝트를 선택하고(Open 열기) 버튼을 클릭합니다.

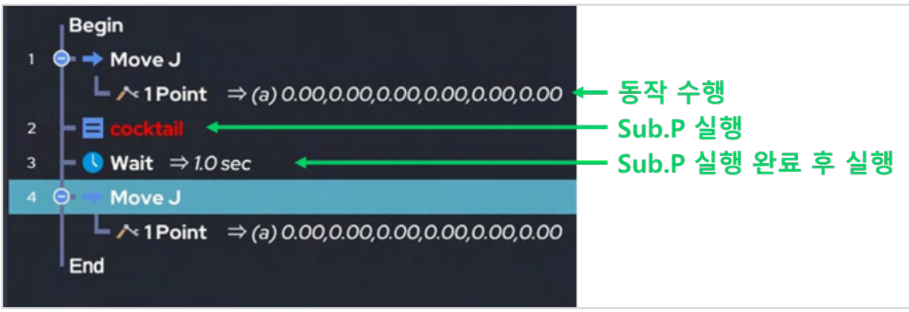
위의 그림과 같이 사전에 만든 프로젝트가, 이번 프로젝트에 삽입되었습니다. 프로그램 Tree 뷰어를 확장형으로 넓게 펼친 후, 불러들인 서브 프로젝트를 클릭하면 화면을 분할하여 불러들인 프로젝트의 내용을 확인할 수 있습니다. 좌측에는 현재 프로젝트가 표기되고, 우측에는 불러온 프로젝트 내용이 표기됩니다.





Sub.P 기능을 통하여 프로젝트도 다른 기능과 같이 시간에 따라 순차적으로 실행됩니다.

Sub.P 기능 다음에 다른 명령어들이 배치된다면, Sub.P 실행이 끝난 이후 해당 명령들이 실행됩니다.



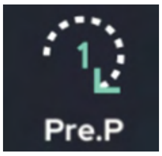
주의

주의:

1) 현재 프로젝트에서 Sub.P 기능으로 호출된 하위 프로젝트는 시각적으로 내용을 볼 수는 있으나, 수정은 불가능합니다. 수정이 필요한 경우 해당 프로젝트를 별도로 열어야 합니다.

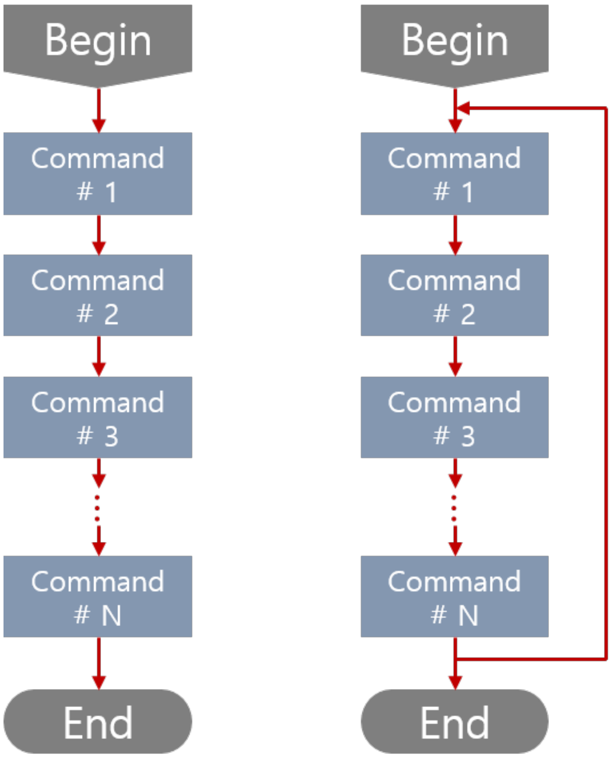
2) Sub.P 기능은 최대 10 단계 깊이까지 호출할 수 있으며, 재귀호출 방식으로 티칭 트리를 구성하는 것을 권장하지 않습니다.

■ Pre.P(Pre Program) 기능:



하위의 내용을 최초 한 번만 실행하는 Pre-Program 기능입니다. Play 모드와 같이, 프로그램이 계속 돌아가는 경우, Pre-Program 의 하위에 선언된 내용은 최초 한 번만 실행됩니다. 변수 선언이나 통신 연결과 같이, 단발성 명령어의 경우 pre-program 하위에서 관리하면 됩니다.

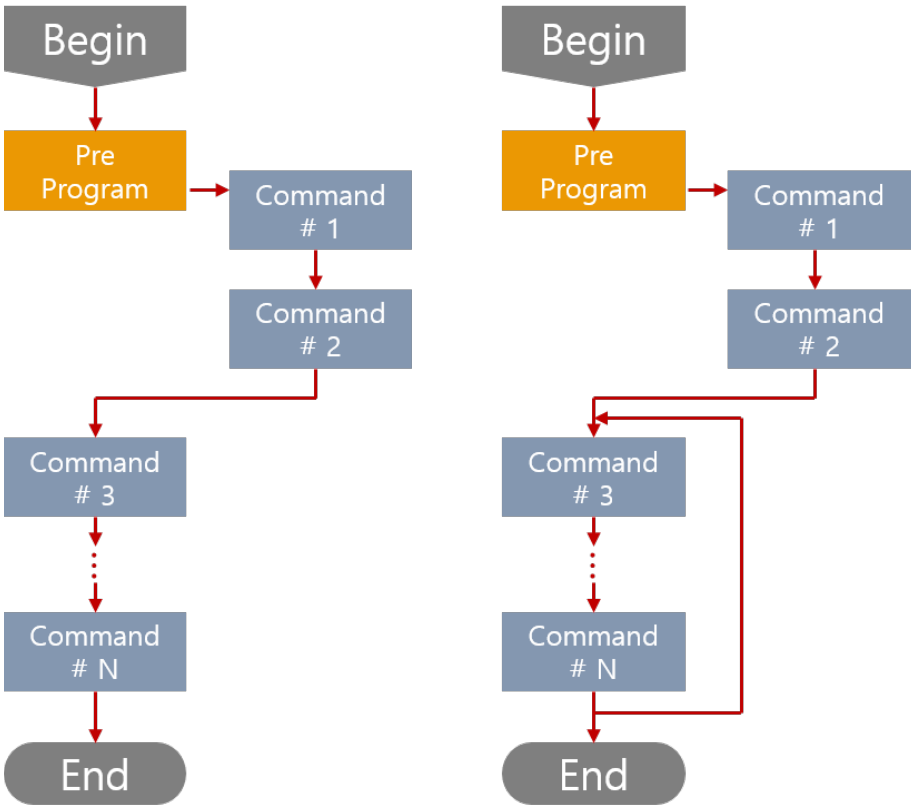
아래의 그림은 Pre.P 기능을 사용하지 않은 경우의 일반적인 명령어 실행 흐름을 보여줍니다.



[Make 화면]   [Play 화면]

좌측은 Make 화면에서 프로그램을 실행한 경우이고, 우측은 Play 화면에서 프로그램을 실행한 경우입니다. Make 에서는 Begin 과 End 사이의 프로그램을 1 회 실행합니다. Play 에서는 Begin 과 End 사이의 프로그램을 반복합니다.

아래의 그림은 Pre.P 기능을 사용한 경우의 프로그램 명령어 실행 흐름을 보여줍니다.



[Make 화면]　　[Play 화면]

Make 화면에서는 Pre.P 의 유무와 상관없이 Begin 과 End 사이의 명령어를 순차적으로 수행합니다. Play 화면에서는 Begin 과 End 사이의 프로그램을 반복하지만, Pre-Program(Pre.P) 하위에 선언된 명령어는 1 회성으로 한 번만 수행합니다.

최초로 한 번만 수행하면 되는 동작이나, 변수 선언, 통신 연결과 같은 기능 및 흐름은 Pre.P 기능을 활용하여 구성하면 됩니다.

아래의 그림은 실제 프로젝트에 Pre.P(Pre Program) 기능이 삽입된 모습입니다. Pre.P 기능은 기능의 특성상 Begin 아래에만 삽입되며, Pre.P 를 통째로 복사해서 다른 곳에 붙여넣기 할 수 없습니다.

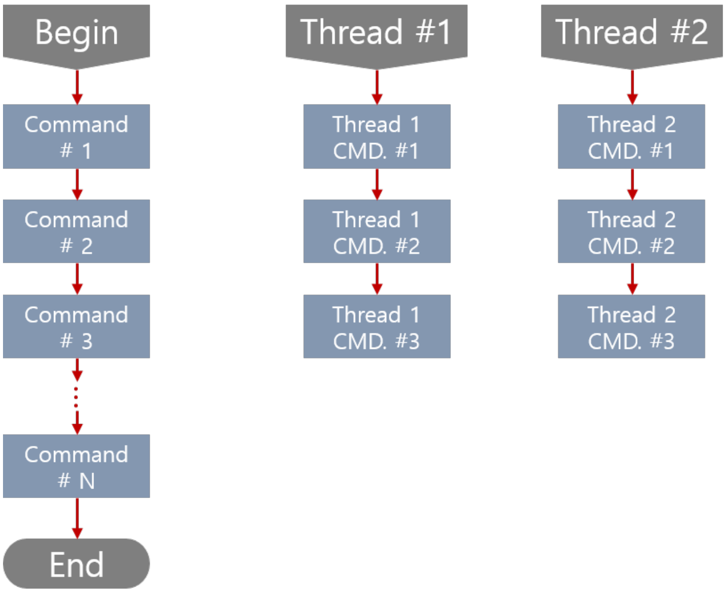
주의

주의:

1) 메인 프로그램에서 Sub.P 기능을 통해 호출된 프로젝트에 Pre.P 기능이 사용되는 경우 Pre.P 기능은 메인 프로그램에서만 적용됩니다.

■ Thread(쓰레드) 기능:

쓰레드 기능입니다. 쓰레드 기능의 하위 항목은 메인 프로그램과 병렬로 실행됩니다. 단, 로봇 동작에 관한 명령어는 쓰레드 하위에 넣지 못합니다.

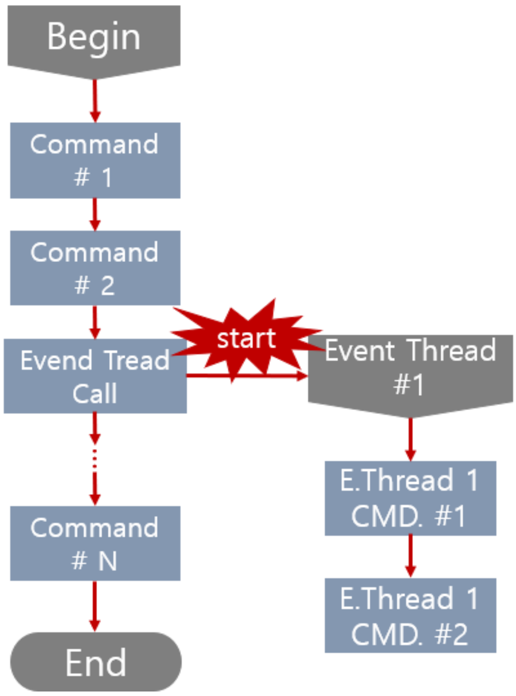


위의 그림과 같이 쓰레드 기능은 메인 프로그램과 병렬로 구성됩니다.

- 쓰레드는 자동으로 반복되지 않습니다. 반복하는 기능을 구현하고자 한다면, 쓰레드 하위에 Repeat(반복문) 기능을 사용하여야 합니다. 만약 1 초 마다 반복되는 쓰레드를 구현하고 싶을 경우, 쓰레드 하위에 Repeat(무한대 옵션)을 넣고, 그 Repeat 하위에 wait(1 sec)를 넣어서 구현합니다.
- 쓰레드 기능은 최대 3 개까지 지원합니다.
- 쓰레드 기능은 최상위 프로그램에서만 작동됩니다. Sub.P 기능을 통해 호출된 하위 프로그램이 쓰레드를 사용할 경우, 정상적으로 작동하지 않습니다.

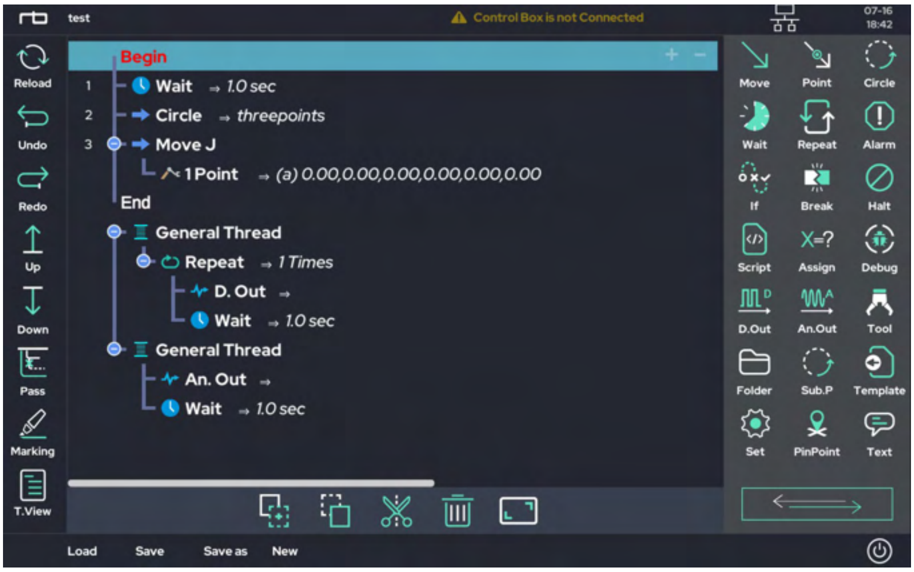
쓰레드의 종류는 다음 설명과 같습니다.

- 일반 쓰레드 : 사용자의 의도적인 일시 정지, 알람, 충돌 감지 등에 의해 같이 정지합니다.
- 비정지 쓰레드 : 충돌 감지 이외에는 정지하지 않습니다.
- 비정지 쓰레드 2 : 프로그램이 종료되기 전까지 정지하지 않습니다.
- 이벤트 일반 쓰레드 : 메인 프로그램에서 이벤트 쓰레드 호출 기능에 의해 실행되는 일반 쓰레드입니다.
- 이벤트 비정지 쓰레드 : 메인 프로그램에서 이벤트 쓰레드 호출 기능에 의해 실행되는 비정지 쓰레드입니다.



위의 그림과 같이 이벤트 쓰레드는 메인 프로그램에서 이벤트 쓰레드 호출 기능을 사용하였을 경우 실행되기 시작합니다.

아래의 모습은 실제 프로젝트에 Thread(쓰레드) 기능을 삽입한 모습을 보여줍니다. 아래의 예시에는 2 개의 쓰레드가 삽입되어 있습니다.



주의

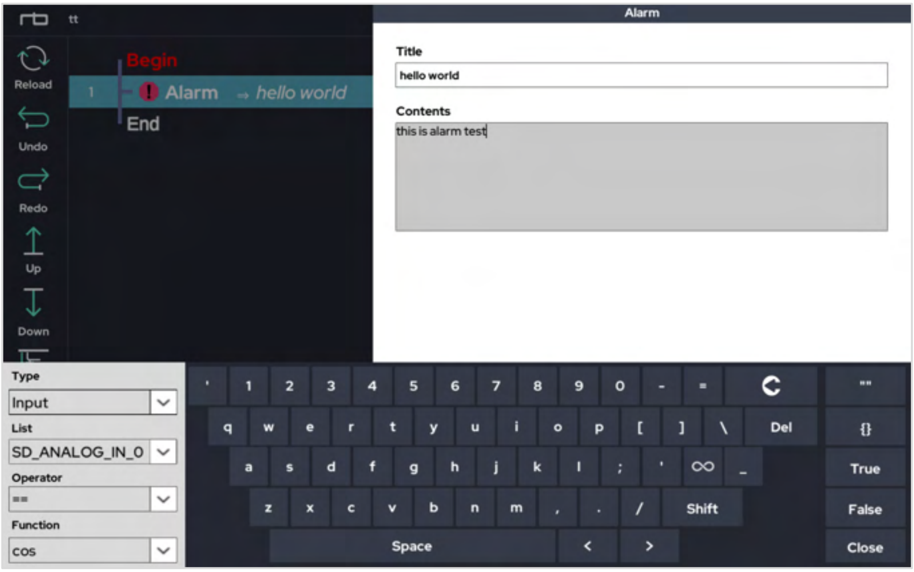
주의:

1) 프로그램의 안정성을 위해 메인 프로젝트 이외에 Sub.P 로 호출되는 프로그램에서는 쓰레드 사용을 권장하지 않습니다.
2) Move 나 Circle 과 같이 로봇 동작에 관한 명령어는 쓰레드 하위에 넣지 못합니다.
3) Pause(일시정지)나 Alarm 기능 사용시 메인 프로그램과 쓰레드, 모두가 일시정지 됩니다. 메인 프로그램이 종료되면 쓰레드도 함께 종료됩니다.

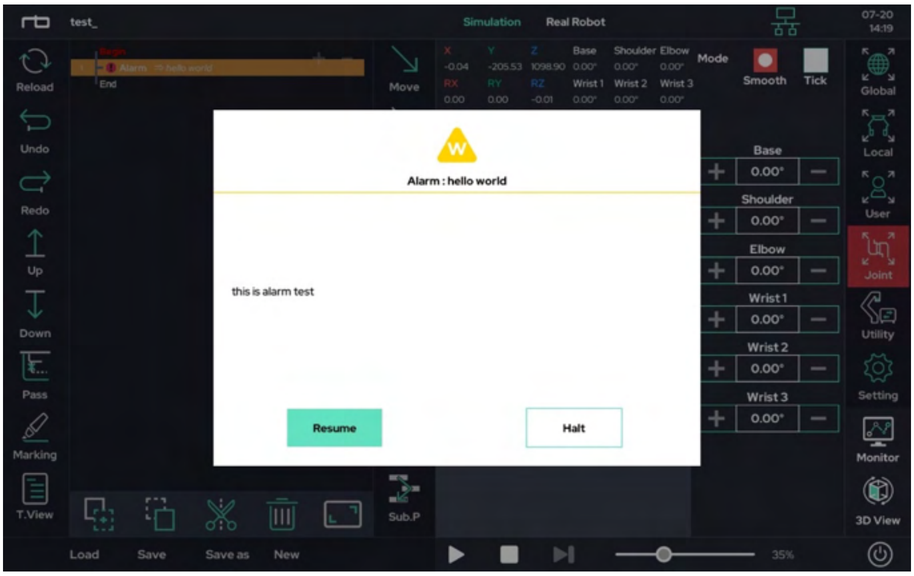
■ Alarm 기능:

알람 (alarm) 팝업 메시지를 띄울 수 있는 기능입니다. 특정 조건이나 구간에 알람 메시지를 넣어서 사용자의 확인을 요청할 수 있습니다.

알람 기능을 프로그램에 삽입 후, 클릭해보면 아래와 같은 설정 창이 나타납니다.



알람 창의 제목과 내용을 사용자가 직접 입력할 수 있습니다. 위와 같이 설정 후, 프로그램을 실행하여 보면, Alarm 명령어가 실행되는 순간 아래와 같은 팝업이 나타납니다.

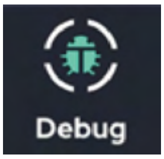


- Resume(계속): 다음 명령어를 계속 수행합니다.
- Halt(정지): 이 시점에서 프로그램을 종료합니다.

Alarm 기능이 수행되는 동안 프로그램 흐름은 일시 정지됩니다. 팝업 창에서 Resume (계속) 버튼을 누르면 다음 명령어를 수행하고, Halt(정지) 버튼을 누르면 이 시점에서 프로그램 구동이 정지됩니다.

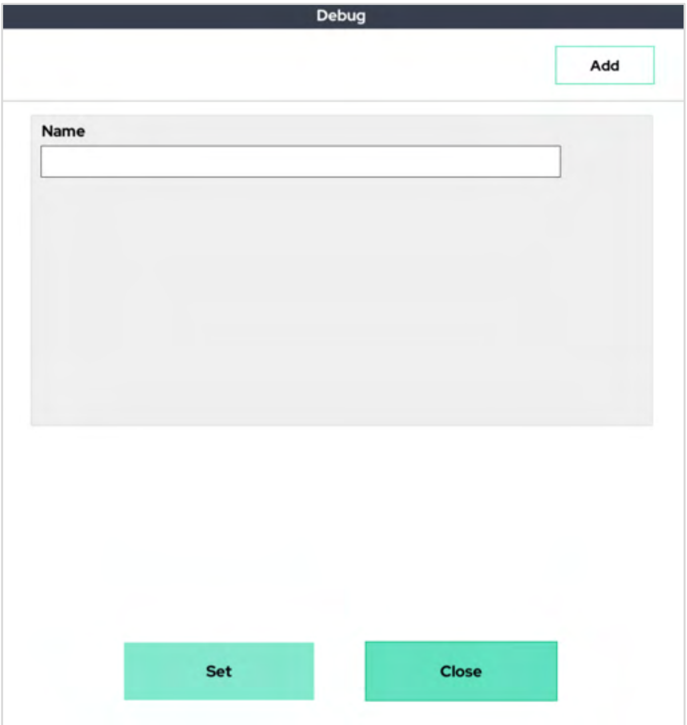
Alarm 기능 사용시 메인 프로그램이 일시정지 되고, 동시에 쓰레드에서 사용중인 흐름도 동시에 일시정지 됩니다.

■ Debug 기능:



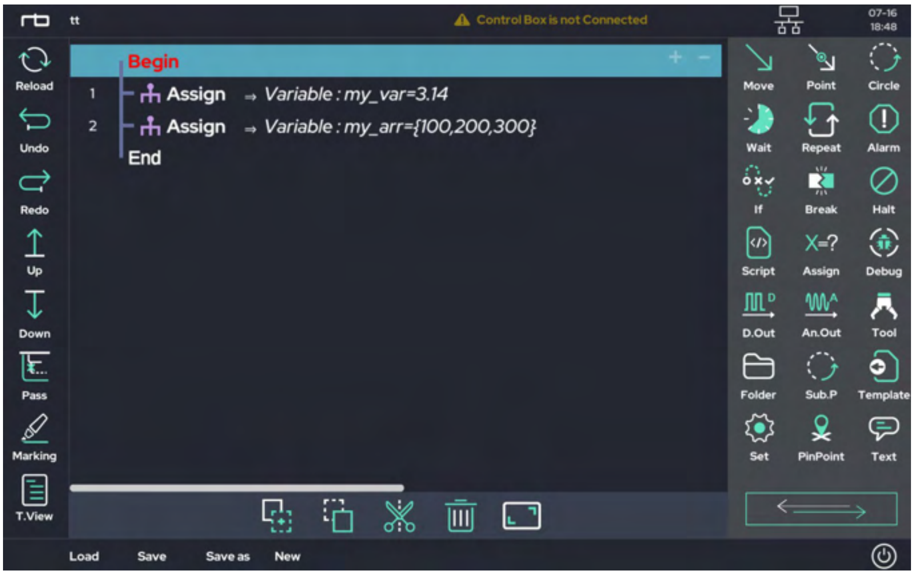
내부 값 디버깅을 위한 Debug 기능입니다. 원하는 변수 값이나 내부 파라미터를 요청하면 팝업 형식으로 해당 값을 확인할 수 있습니다.

디버깅 기능은 내부의 변수를 관찰하기 위한 기능입니다. 주로 프로그램 티칭/개발시에 프로그램에서 사용중인 변수 값을 확인하는 용도로 사용됩니다.



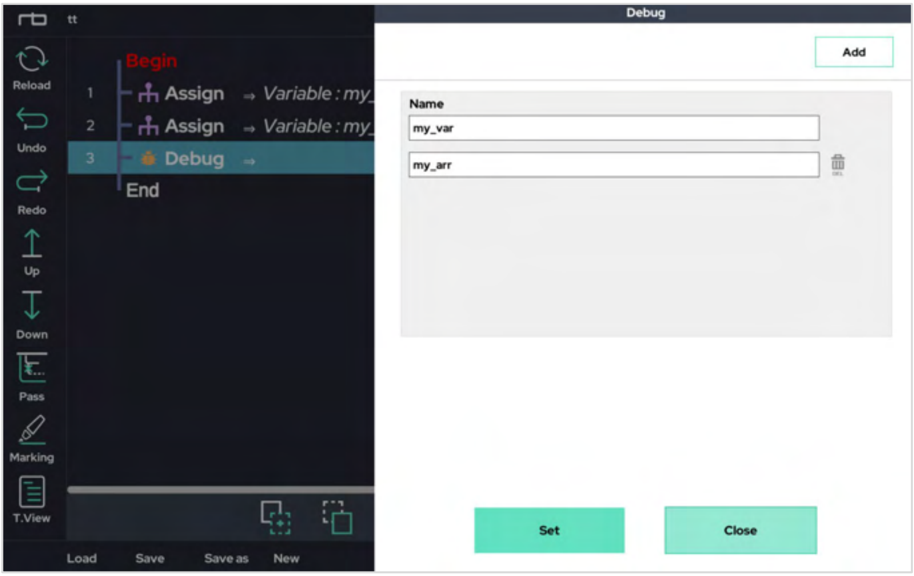
디버그 기능을 프로그램 Tree 에 추가한 후, Debug 를 클릭하여 보면 위와 같은 팝업 창이 뜹니다. 이름(Name) 항목에 원하는 변수명을 넣으면 됩니다. 만약 여러 변수를 동시에 관찰하고 싶을 경우(Add) 버튼을 눌러서 항목을 추가합니다.

예시를 통해 디버그 기능을 알아보겠습니다. 아래와 같이 Assign 기능을 통해 하나의 variable 타입 변수(my_var = 3.14)와 하나의 array 타입 변수(my_arr = {100,200,300})를 선언합니다.

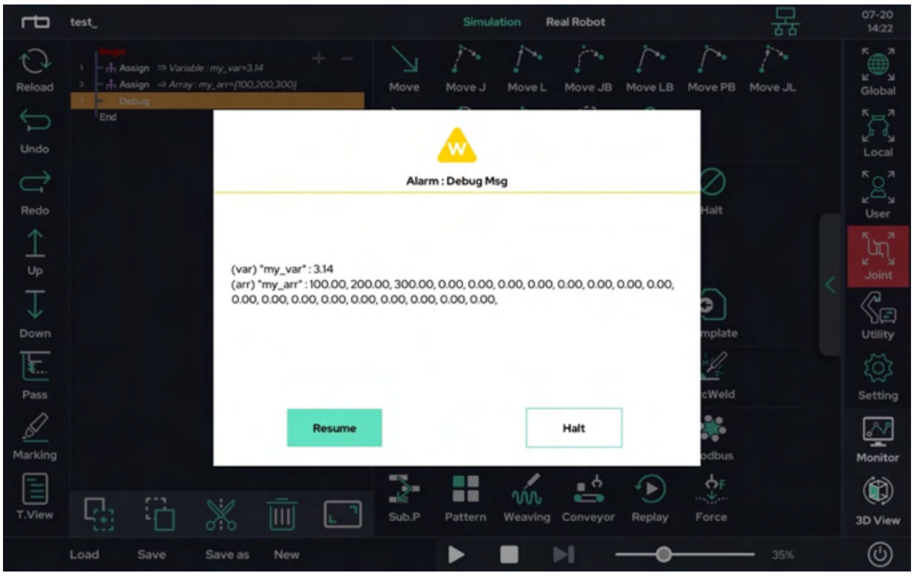


그 아래에 Debug 기능을 추가합니다.

Debug 속성창에서 아래와 같이 앞서 선언된 두 변수를 관찰한다고 설정합니다.

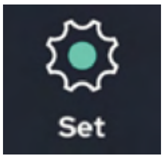


설정을 완료 후 프로그램을 실행(실행 전 태블릿 PC 와 컨트롤 박스는 연결되어야 합니다)하면, Debug 명령어가 실행되는 순간 아래와 같은 팝업창이 뜨면서 지정한 변수 값들을 관찰할 수 있습니다.



- Resume(계속): 다음 명령어를 계속 수행합니다.
- Halt(정지): 이 시점에서 프로그램을 종료합니다.

■ Set 기능:



Set 기능입니다. Setup 메뉴에서 세팅하는 각종 설정 파라미터를 프로그램 내부에서 별도로 세팅 가능합니다. 특정 설정 파라미터를 프로그램 실행 중에 변경할 수 있는 기능을 가지고 있습니다.

Setup 설정 화면에서 변경하는 각종 파라미터들은 해당 컨트롤 박스를 사용하는 모든 프로젝트에 기본 값으로 적용됩니다. 만약 특정 프로젝트에 특정 파라미터를 별도로 사용해야 한다면 Set 기능을 프로젝트 상단에 추가하여(예: Pre.P 하위) 프로젝트 별로 파라미터 설정을 관리할 수 있습니다.

Set 기능은 영구 세팅이 아닌 임시 세팅입니다. 같은 파라미터 설정에 대하여 새로운 Set 기능이 호출되면 새로운 Set 기능을 기준으로 파라미터가 반영됩니다.

프로그램이 종료되면, Setup 에서 설정한 기본 값으로 파라미터 세팅이 돌아옵니다.

Set 기능에서 제공되는 하위 기능은 아래와 같습니다.

- 타이머 리셋
- 충돌 감지 감도 조절
- TCP 하중 설정
- 동작 오프셋
- 인박스 기능 설정
- TCP 위치 임시 설정
- 툴 안전 영역 설정
- 작업 안전 영역 설정
- 인박스 영역 설정
- 충돌 감지 On/Off
- 속도 배율 일괄 조정
- 가속도 배율 일괄 조정
- 시리얼 통신 설정
- 동작 속도 직접 설정

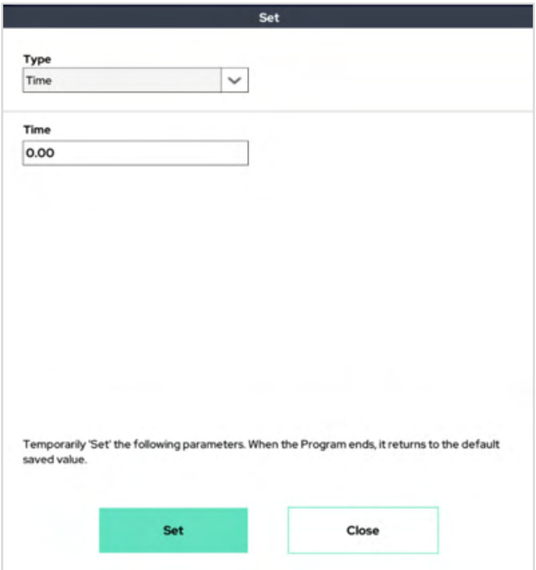
- 나선 원형 동작 설정
- 속도 조절 바 제어
- 충돌 감지 후 로봇 정지 모드
- 사용자 좌표계 시프팅
- 충돌 감지 후 프로그램 흐름
- 제어박스 출력 임시 제어
- L 계열 동작 XYZ 위치 값 고정
- L 계열 동작 회전 값 고정
- 사용자 좌표계 임시 설정
- 목표 지점 위치 값 시프트
- 목표 지점 위치 값 시프트 2
- 진동 센서 감지 On/Off
- 디지털 입력 신호 시뮬레이션
- 프로그램 흐름 제어
- 고속 가속도 모드
- 동작 시간 구속
- 고 민감도 충돌감지
- 오프셋 미세 조정
- 사용자 좌표계 시프팅 6D
- 사용자 좌표계 자동 정렬
- 타이머 설정
- 노아크 동작 속도

주의

주의:

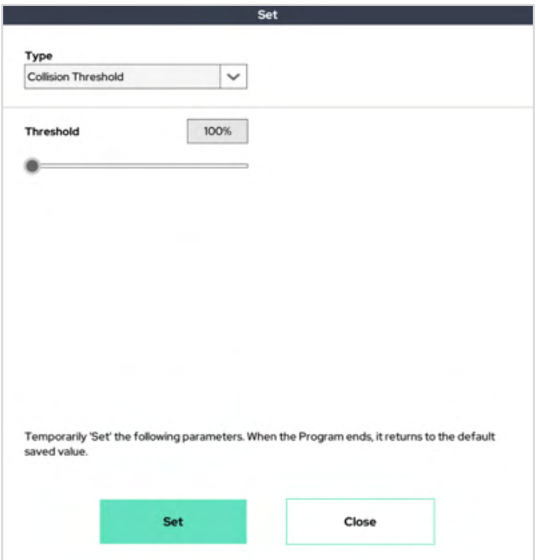
1) Set 기능에서 설정된 값은 임시 값입니다. 프로그램이 종료되면 Setup 에서 설정된 기본 값으로 자동으로 돌아옵니다.
2) Set 기능에서 제공되는 기능들은 프로그램 흐름 중간에서 다른 값으로 변경 가능합니다. 예를 들어 Set 의 'Collision On/Off'기능을 사용하면, 프로그램 흐름 중간중간에 충돌 감지를 끄고 싶을 경우 선택적으로 끌 수 있습니다.

Set 기능: Time(타이머 리셋)



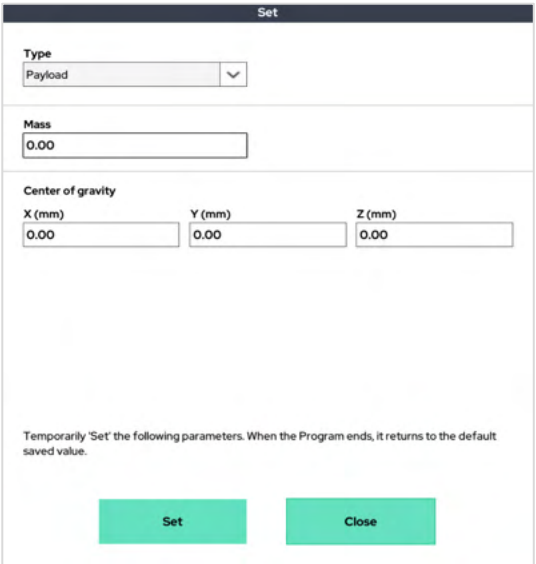
타이머를 시작하며 초기 시작 값을 설정합니다. 입력한 값부터 타이머의 값이 증가합니다.

Set 기능: Collision Threshold(충돌 감지 감도 조절)



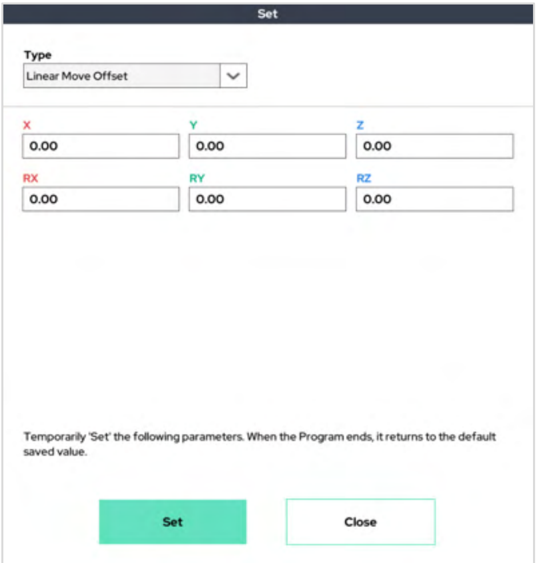
충돌 감지 감도를 임시 설정합니다. 낮을수록 민감한 감지 기능으로 설정됩니다. Setup-Cobot 에서 설정하는 것과 같은 물리적 의미를 갖습니다.

Set 기능: Tool Payload(TCP 하중 설정)



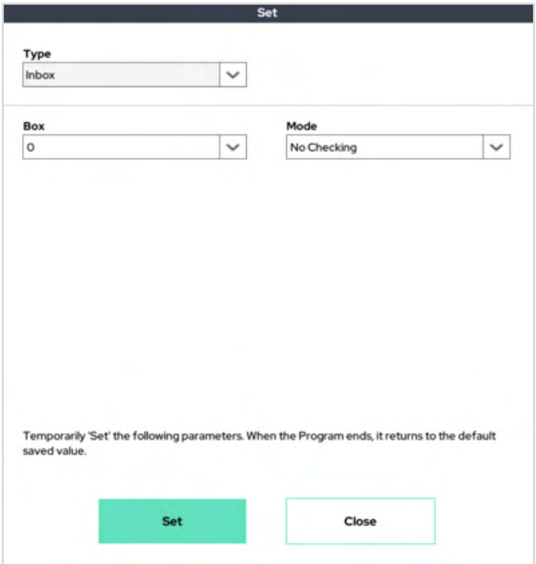
툴의 무게와 무게 중심을 임시 설정합니다. Setup-Tool 에서 설정하는 것과 같은 물리적 의미를 갖습니다.

Set 기능: Linear Move Offset(동작 오프셋)



베이스 좌표계를 기준으로 약간의 오프셋을 줄 수 있습니다. 이 기능을 통해 최대 20 mm 의 오프셋 임시 설정이 가능합니다.

Set 기능: Inbox(인박스 기능 설정)

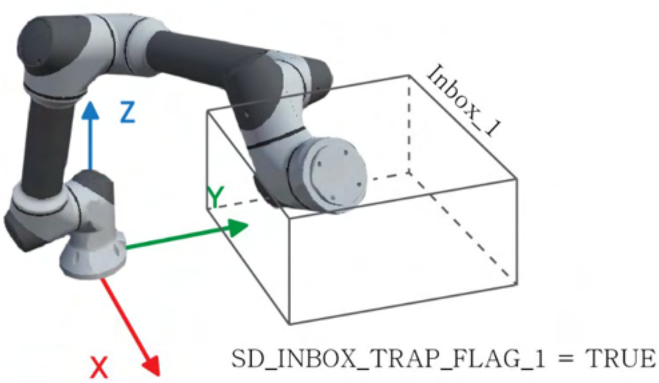


Inbox 기능의 옵션을 선택합니다. Inbox 기능은 로봇의 특정 부위가 특정 영역 안에 들어와 있는지 아닌지를 자동으로 체크해주는 기능입니다(Worldzone Trap 기능). 체크할 수 있는 부위는 아래와 같습니다.

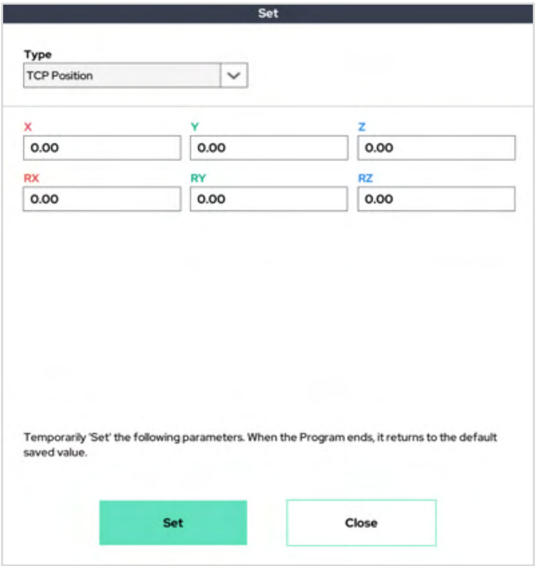
로봇 팔 말단의 툴 플랜지 중심이 특정 영역에 들어왔는지 확인.

- 로봇 팔 말단의 설정된 TCP 가 특정 영역에 들어왔는지 확인.
- 로봇 팔 말단에 가상으로 설정한 Box 가 특정 영역에 들어왔는지 확인.
- 위의 3 경우 중 하나라도 특정 영역에 들어왔는지 확인.

특정 영역을 설정하는 Inbox 의 크기와 위치는 Setup-Inbox 에서 설정할 수 있습니다. 위의 설정을 바탕으로 특정 영역에 들어와 있는지 여부는 Shared Data 타입의 데이터 중 SD_INBOX_TRAP_FLAG_0 또는 SD_INBOX_TRAP_FLAG_1 을 통해 사용할 수 있습니다.

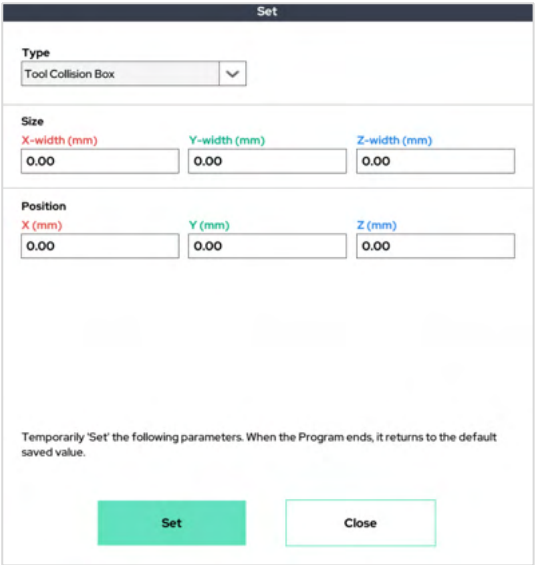


Set 기능: TCP Position(TCP 위치 임시 설정)



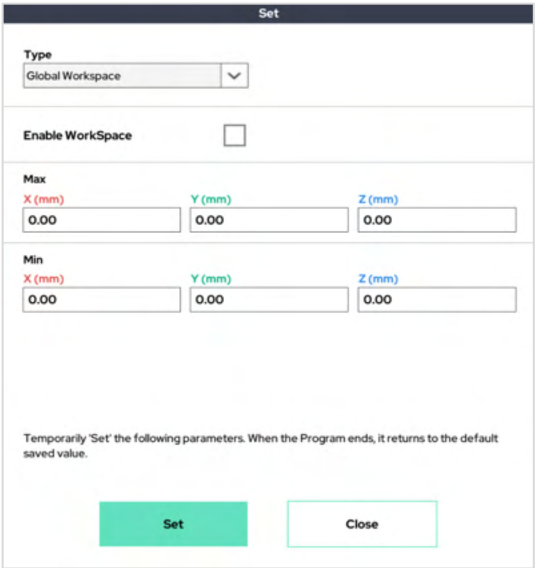
툴의 TCP 위치와 툴 좌표계를 임시 설정합니다. Setup-Tool 에서 설정하는 것과 같은 물리적 의미를 갖습니다.

Set 기능: Tool Collision Box(툴 안전 영역 설정)



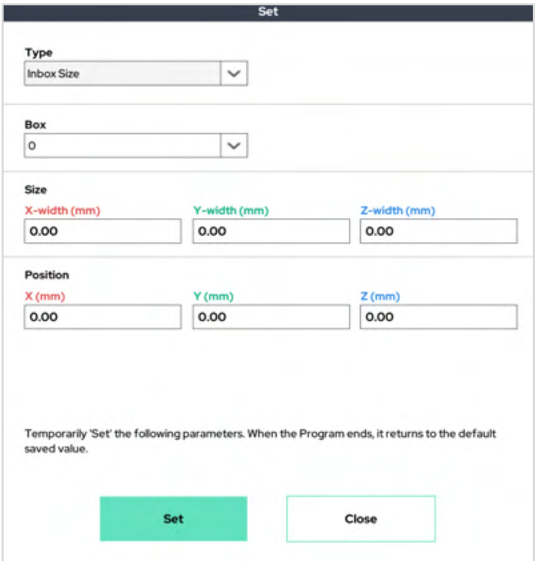
자가 충돌 방지 기능을 위한 툴 가상 박스의 크기와 위치를 임시 설정합니다. Setup-Tool 에서 설정하는 것과 같은 물리적 의미를 갖습니다.

Set 기능: Global Workspace(작업 안전 영역 설정)



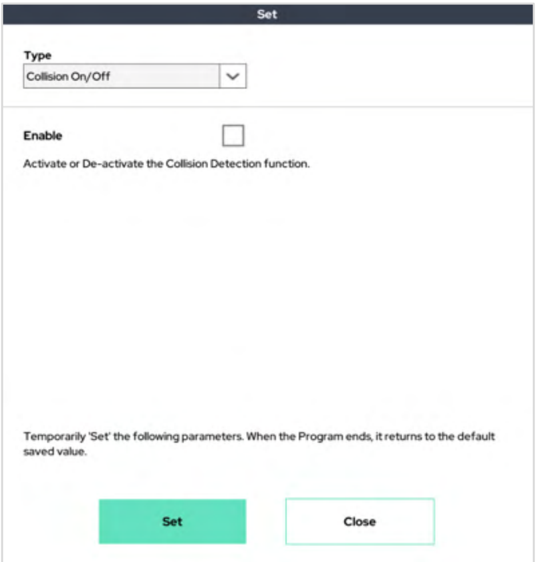
자가 충돌 방지 기능을 위한 Workspace 공간의 범주를 임시 설정합니다. Setup-Cobot 에서 설정하는 것과 같은 물리적 의미를 갖습니다.

Set 기능: Inbox Size(인박스 영역 설정)



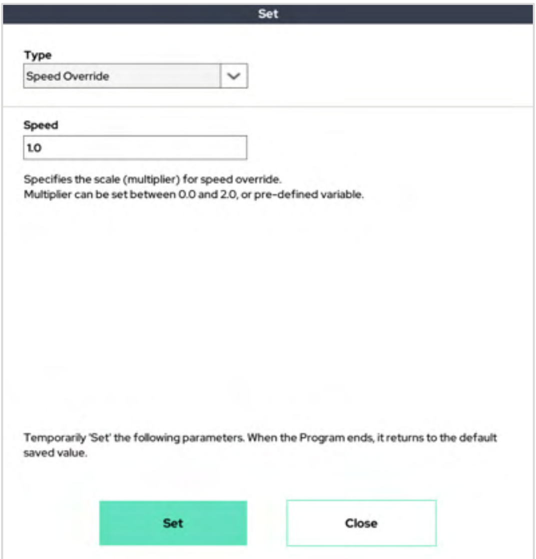
Inbox 의 위치와 크기를 임시 설정합니다. Setup-Inbox 에서 설정하는 것과 같은 물리적 의미를 갖습니다.

Set 기능: Collision On/Off(충돌 감지 On/Off)

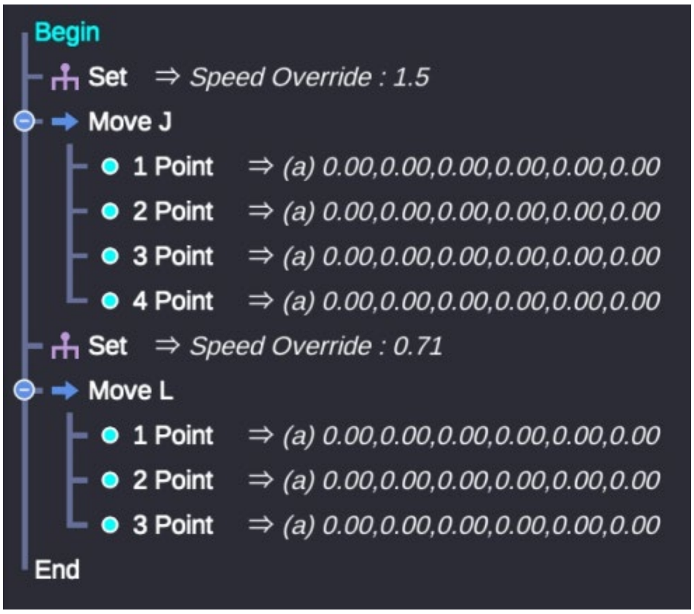


외부 충돌 감지 모드의 사용 유무를 임시 설정합니다. Setup-Cobot 에서 설정하는 것과 같은 물리적 의미를 갖습니다.

Set 기능: Speed Override(속도 배율 일괄 조정)

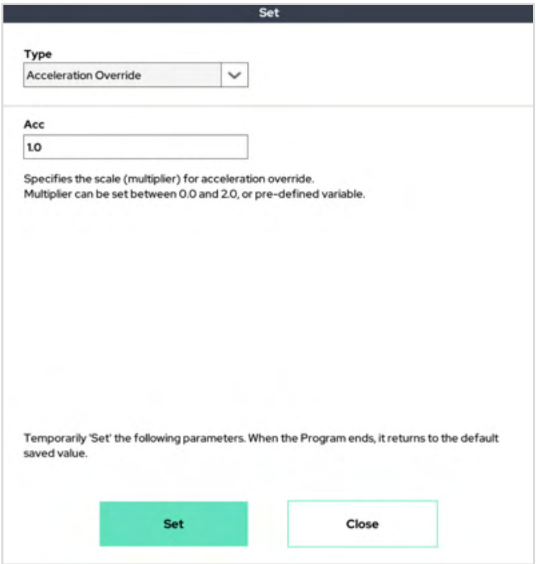


티칭 시 Move 기능과 Point 기능을 통해 설정된 속도 값을 한꺼번에 바꾸는 기능입니다. 0~2.0 사이 값을 넣을 수 있으며, 변수화된 정보를 넣어도 됩니다.



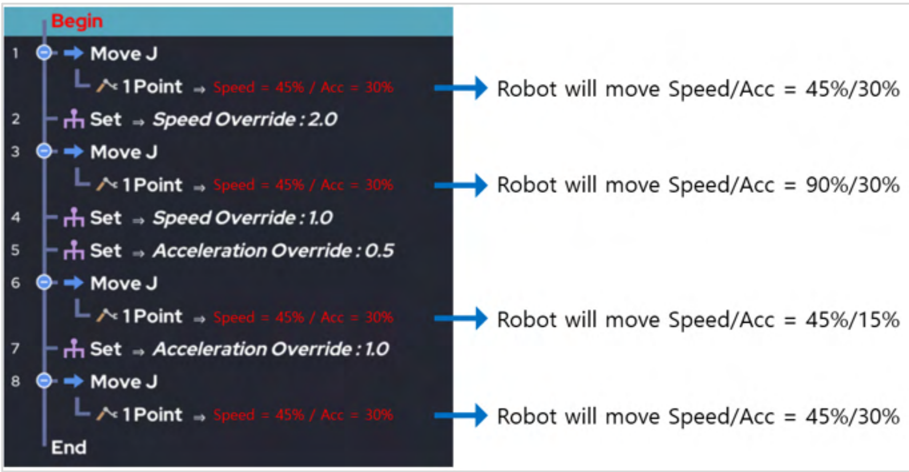
위의 예시에서는 Move J 앞에서 Set-Speed-Override 1.5 가 되어 있습니다. 따라서 Move J 쪽 동작에서는 설정한 속도의 1.5 배의 속도가 설정된 것과 같은 효과로 동작합니다. Move J 와 Move L 사이에는 Set-Speed-Override 0.71 이 사용되었습니다. 따라서 Move L 동작에서는 설정한 속도의 0.71 배의 속도가 설정된 것과 같은 효과로 동작합니다.

Set 기능: Acceleration Override(가속도 배율 일괄 조정)

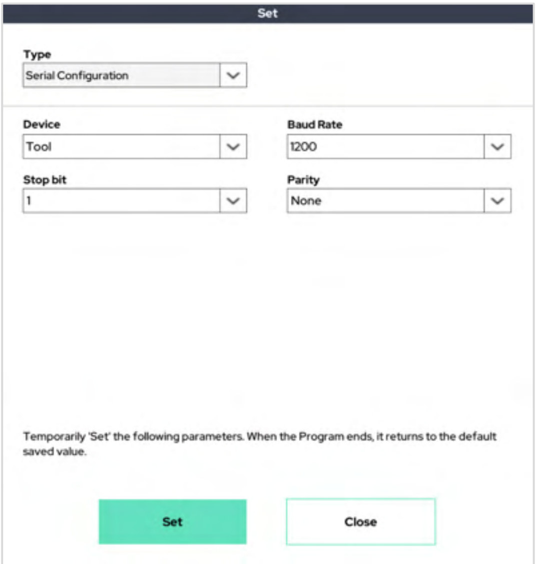


티칭 시 Move 기능과 Point 기능을 통해 설정된 가속도 값을 한꺼번에 바꾸는 기능입니다. 0~2.0 사이 값을 넣을 수 있으며, 변수화된 정보를 넣어도 됩니다.

아래 코드를 통해 Speed Override, Acceleration Override 를 사용할 경우 속도와 가속도가 어떻게 변화하는지 확인할 수 있다.

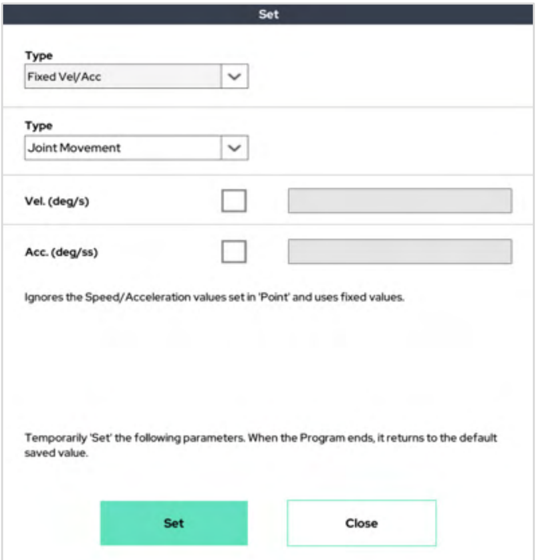


Set 기능: Serial Configuration(시리얼 통신 설정)



시리얼 통신의 통신 속도 (Baud Rate)와 규약 (Stop bit / Parity)을 임시 설정합니다. Setup-Serial 에서 설정하는 것과 같은 의미를 가집니다.

Set 기능: Fixed Vel/Acc(동작 속도 직접 설정)



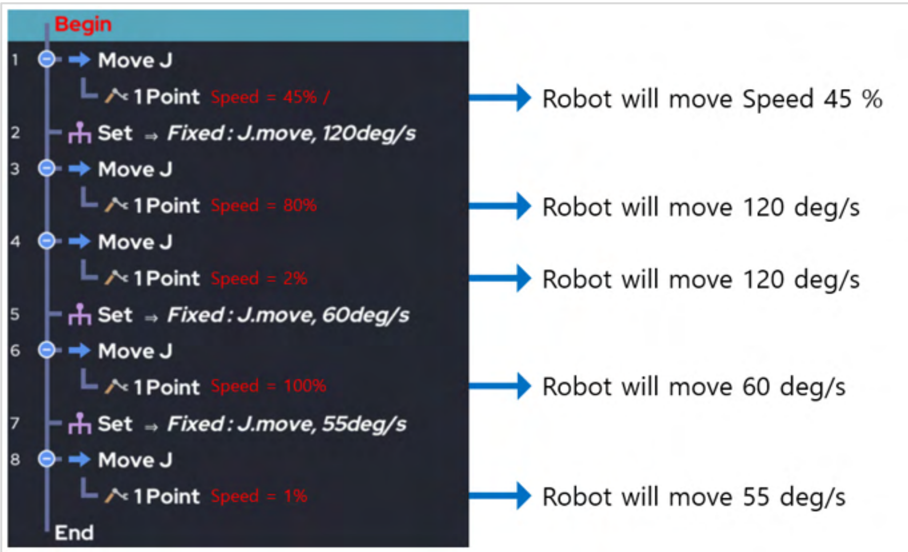
Move 의 Point 마다 설정된 속도/가속도를 무시하고, 고정된 값을 사용하고 싶을 때 사용하는 기능입니다. Joint Movement 와 Linear Movement, 두 가지 하위 옵션이 존재합니다.

Joint Movement 에서 설정된 속도 (deg/s)와 가속도 (deg/s^2)는 Joint movement type 인 MoveJ 와 MoveJB 의 동작 속도와 가속도에 영향을 미칩니다.

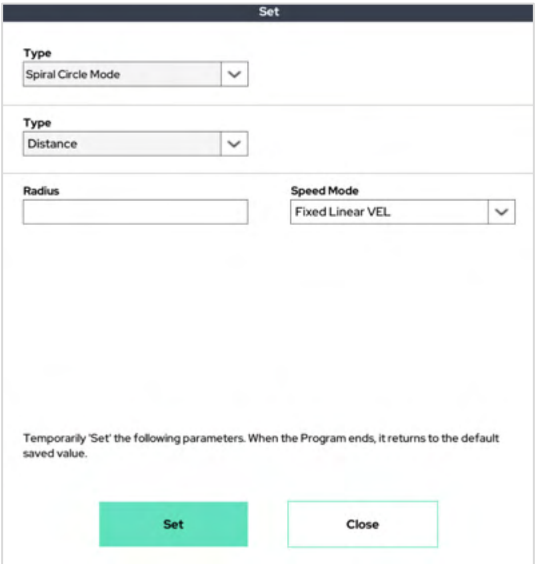
Linear Movement 에서 설정된 속도 (mm/s)와 가속도 (mm/s^2)는 Linear movement type 인 MoveL, MoveLB, MovePB, MoveJL, MoveITPL 그리고 Circle 의 동작 속도와 가속도에 영향을 미칩니다.

이 기능을 통해서 속도/가속도를 강제하고 싶지 않을 경우 체크 박스를 해제하면 됩니다. 이 경우, 동작 시 Point 마다 설정된 속도/가속도 값을 따릅니다.

예) 동작 시 특정 속도와 가속도를 지켜야 하는 경우 이 Set 기능을 아래 코드와 같이 사용하면 됩니다.

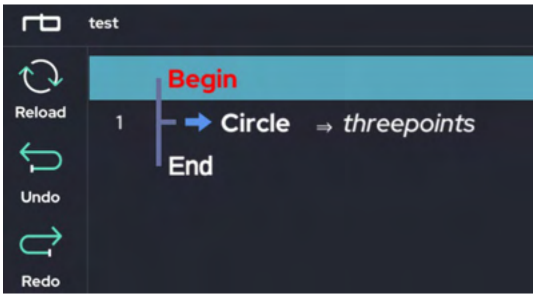


Set 기능: Spiral Circle Mode(나선 원형 동작 설정)



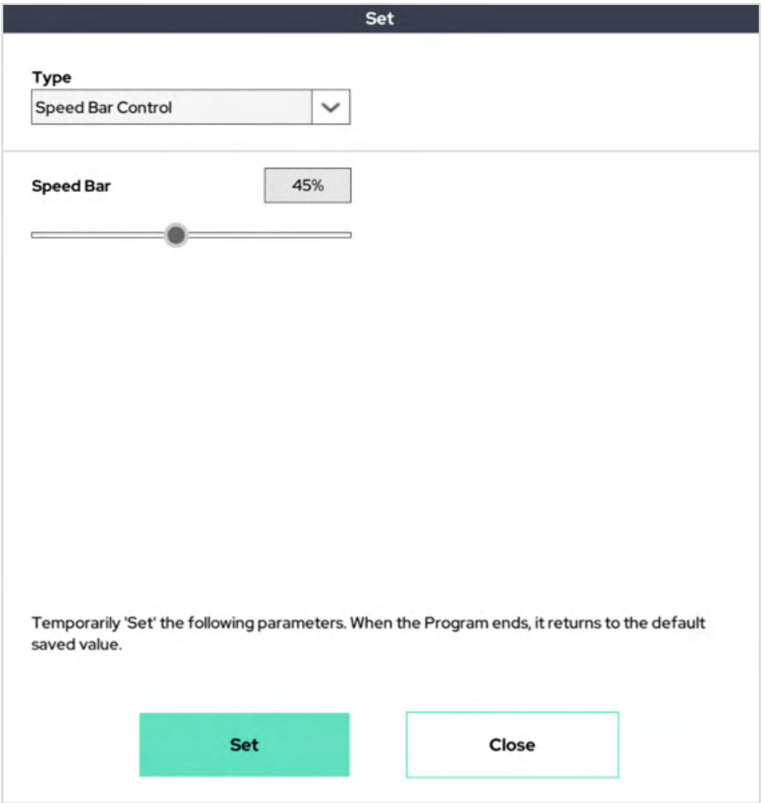
원형 동작을 나선형 (Spiral) 동작으로 바꾸기 위한 기능입니다. Circle 기능만 사용할 경우 원/호를 그립니다. Circle 기능 위에 Set-Spiral mode 가 사용된다면, 기존의 원/호는 나선형 동작으로 변경됩니다. 따라서 나선형 동작을 구현하기 위해서는 이 기능을 Circle 기능 위에 삽입해야 합니다.

예1) Circle 만 사용된 경우: 일반적인 원/호 궤적을 만듦.



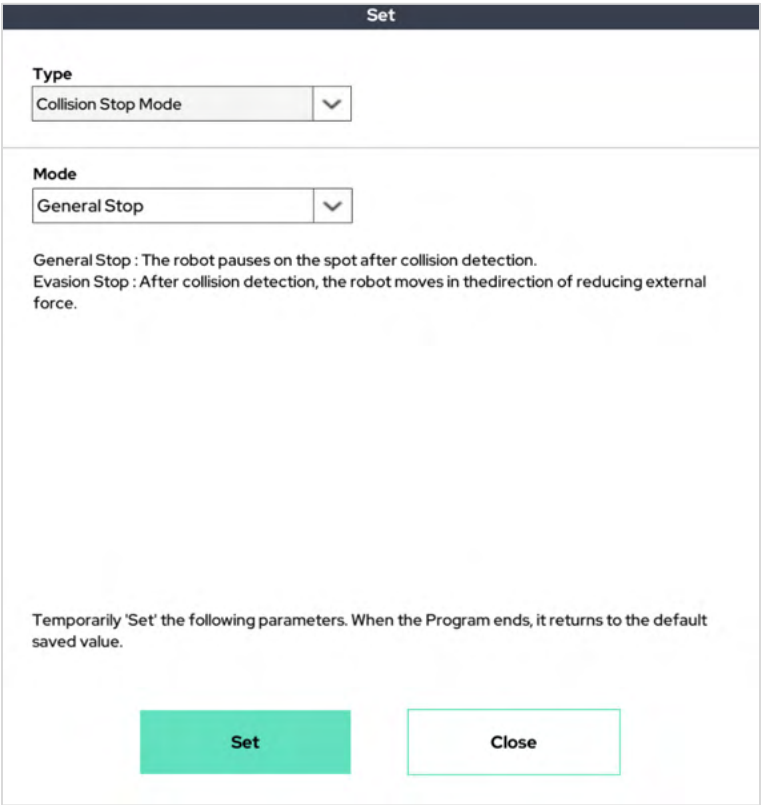
예 2) Set-SpiralMode + Circle 기능: 나선형 궤적을 만듦.

Set 기능: Speed Bar Control(속도 조절 바 제어)



UI 의 속도 조절 바 (화면 하단 우측)를 프로그램으로 조절할 수 있는 기능입니다. 원하는 구간에서 이 기능을 사용함으로써 UI 속도 조절 바를 바꿀 수 있습니다.

Set 기능: Collision Stop Mode(충돌 감지 후 로봇 정지 모드)



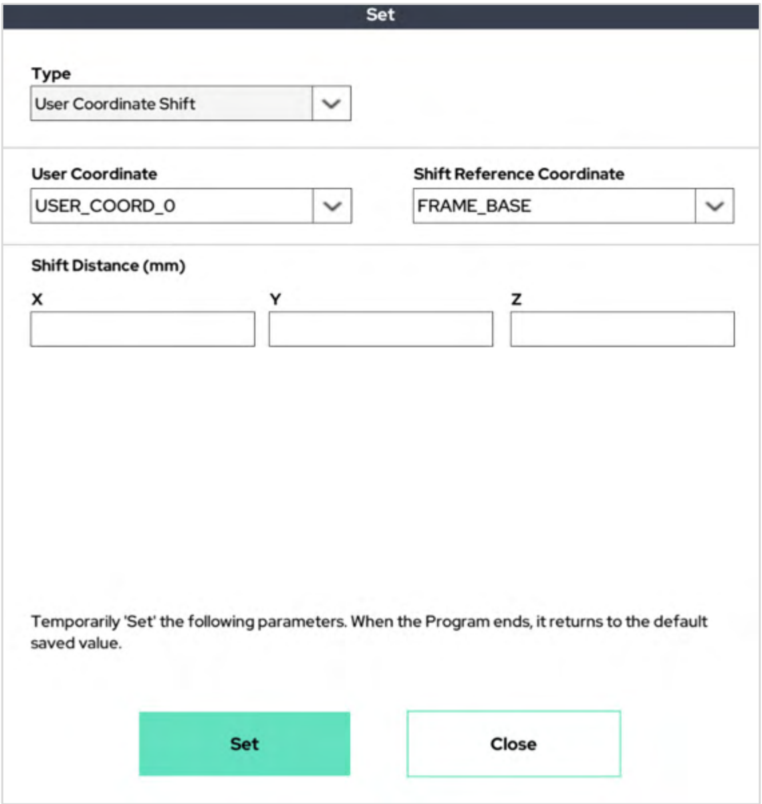
외부 충돌 감지 시 로봇의 동작 유형을 선택합니다. 두 가지 옵션이 있습니다.

General Stop: 충돌 감지 후, 그 자리에서 궤적 움직임을 일시 정지합니다.

Evasion Stop: 충돌 감지 후, 외부 힘을 피하는 방향으로 로봇이 조금 이동 후 궤적 움직임을 일시 정지합니다.

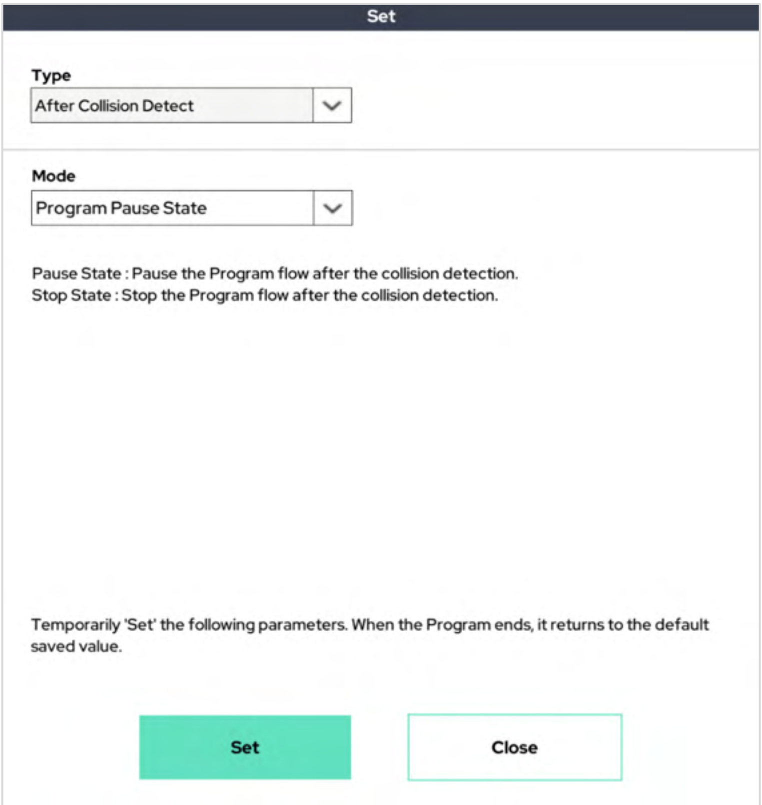
Setup-Cobot 의 '충돌 감지 보호모드'(Action after Collision)와 같은 물리적 의미를 가집니다.

Set 기능: User Coordinate Shift(사용자 좌표계 시프팅)



사용자 좌표계의 원점을 임시로 이동 (Shift)할 수 있는 기능입니다. Shift 시키고자 하는 사용자 좌표계 번호와 Shift 거리를 설정하고, Shift 거리를 어떤 좌표계를 기준으로 움직일 지를 선택할 수 있습니다.

Set 기능: After Collision Detect(충돌 감지 후 프로그램 흐름)



외부 충돌 감지 후 프로그램 흐름을 선택할 수 있습니다.

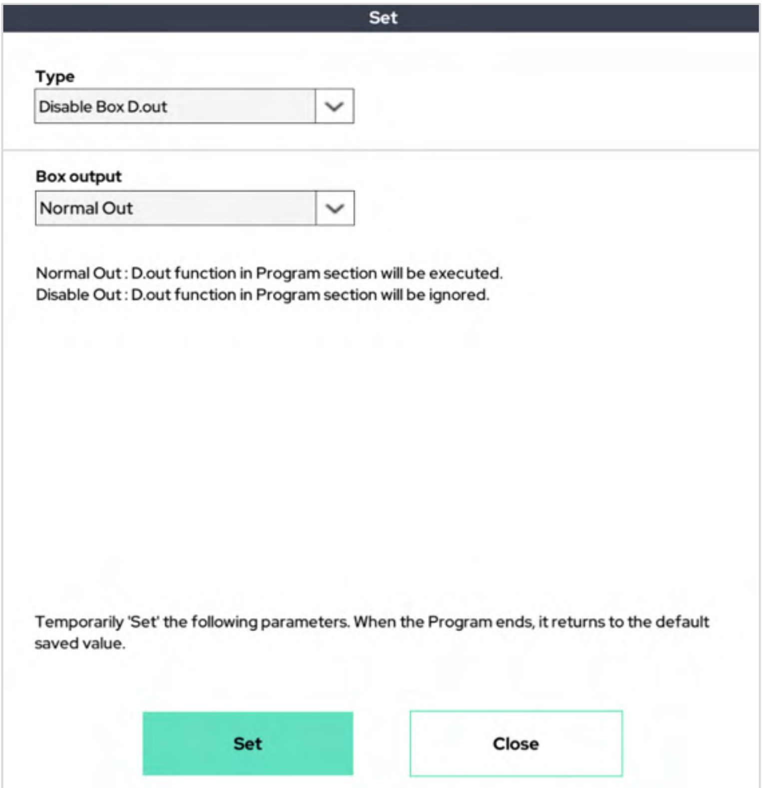
당사의 기본 설정은 외부 충돌 감지 후, 프로그램이 일시정지 하는 형태입니다. 충돌 감지 후 충돌 감지 알림창이 뜨며 프로그램 및 쓰레드가 일시정지되는 것이 기본 형태입니다.

만약, 충돌 감지 후, 프로그램을 종료하고 싶다면 이 기능을 사용하여 옵션을 Stop state 로 선택하면 됩니다.

Pause State: 외부 충돌 감지 후 프로그램 흐름이 일시정지 됩니다.

Stop State: 외부 충돌 감지 후 프로그램 흐름이 정지됩니다.

Set 기능: Disable Box D.out(제어박스 출력 임시 제어)

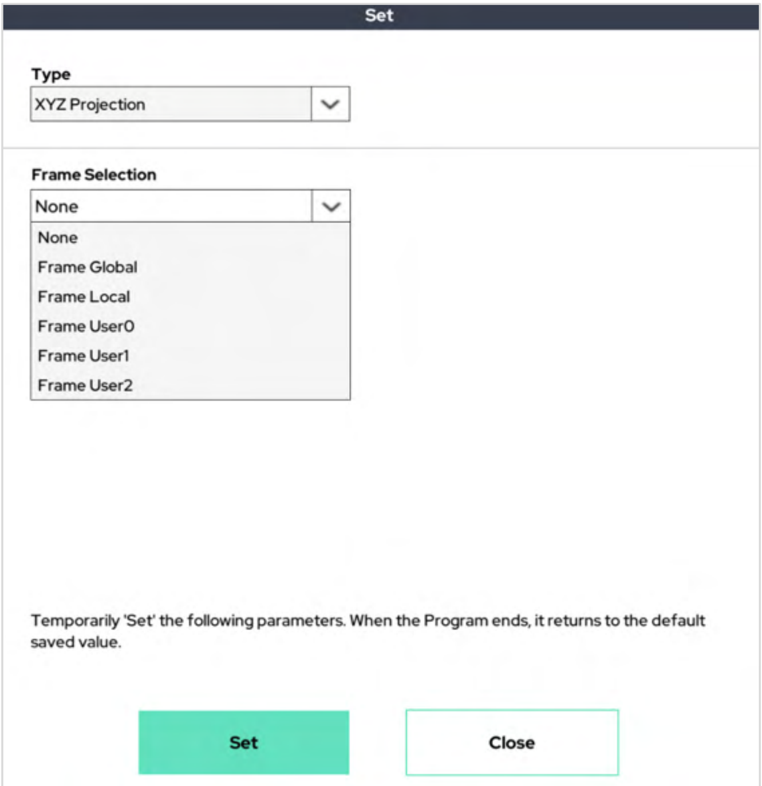


임시로 제어 박스의 디지털 출력을 비활성화 하는 기능입니다.

프로그램에 삽입한 디지털 출력 명령어를 지우지 않아도, 이 Set 명령어를 사용하면, 특정 구간의 디지털 출력 명령을 무시할 수 있습니다.

개발 테스트 등의 목적으로 사용 가능하며 옵션 선택을 통하여, 프로그램 구간에 따라 출력을 비활성/활성 시킬 수 있습니다.

Set 기능: XYZ Projection(L 계열 동작 XYZ 위치 값 고정)

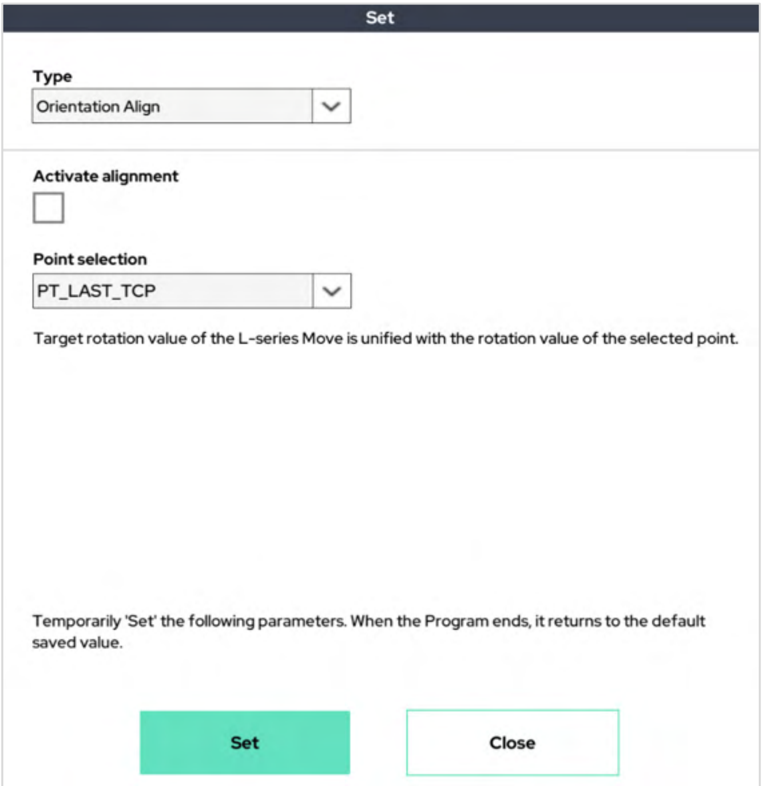


L 계열 동작 (예: MoveL. MovePB, Circle etc)의 목표 위치 좌표 값을 고정시키는 기능입니다. 고정시킬 값과 기준 좌표계를 선택하면, 목표 지점 또는 설정 지점의 위치 좌표 값이 선택한 좌표계의 선택된 축의 값으로 고정됩니다.

예를 들어, 좌표계를 베이스 좌표계 (Global)을 선택하고, Z Projection 100mm 를 선택/기입할 경우, 모든 이동 목표값/설정 좌표 값들의 Z 높이는 100mm 로 일괄 적용됩니다.

이 기능 또한 Set 의 기능으로써 프로그램 구간 구간 마다 활성화/비활성화 가능합니다. 비활성화 하고 싶을 경우 좌표계에서 None 을 선택하면 됩니다.

Set 기능: Orientation Align(L 계열 동작 회전 값 고정)

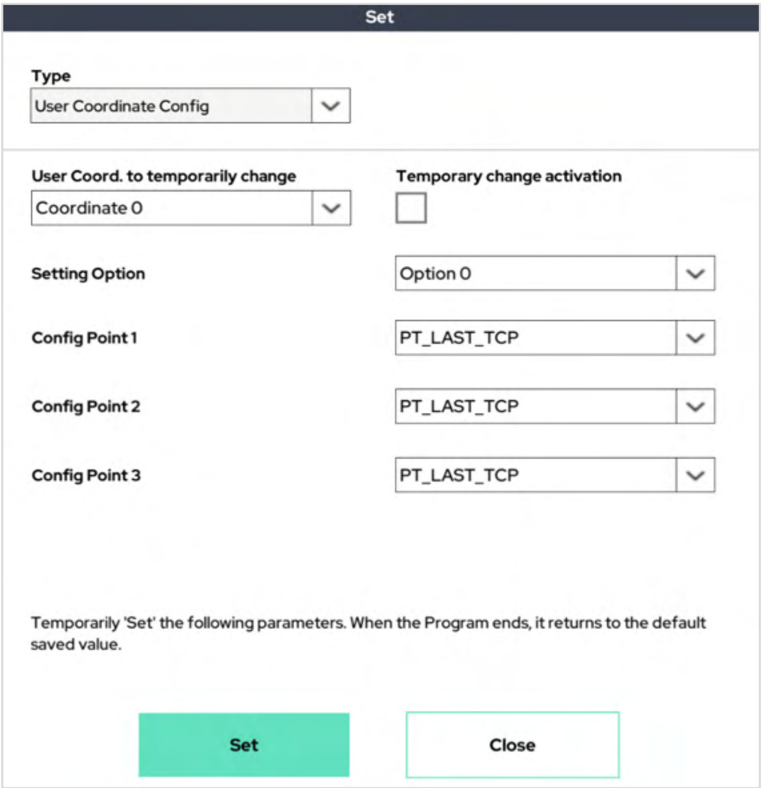


L 계열 동작 (예: MoveL. MovePB, Circle etc)의 목표 회전 좌표 값을 고정시키는 기능입니다.

선택한 Point 의 회전 값으로 L 계열 동작들의 회전을 고정시킵니다.

Set 기능의 하위 기능으로써 프로그램 구간에 따라 이 기능을 켜거나 끌 수 있으며, TCP 의 회전을 일정하게 한 번에 통일시키고 싶을 때, 사용할 수 있는 기능입니다.

Set 기능: User Coordinate Config(사용자 좌표계 임시 설정)

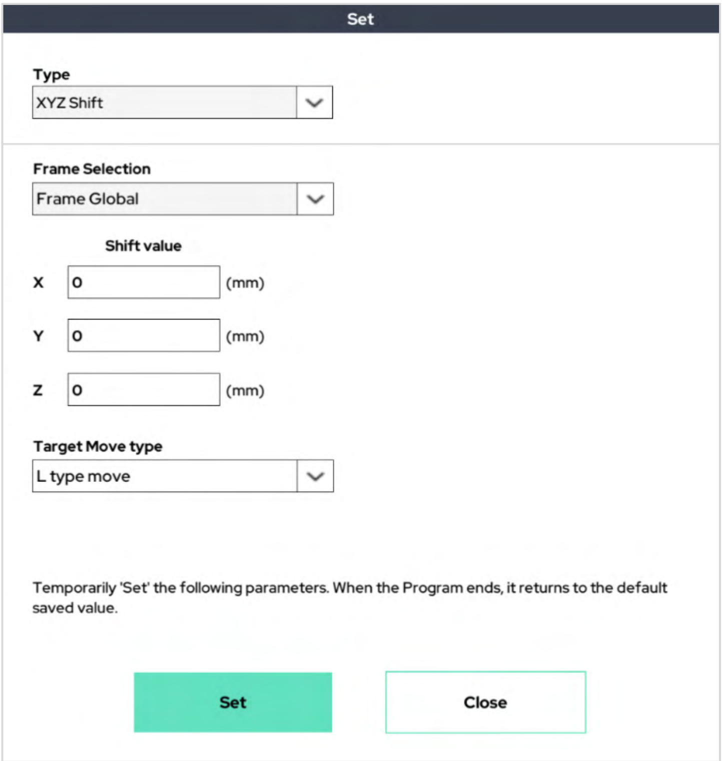


사용자 좌표계 설정을 임시로 바꿀 수 있는 기능입니다.

프로그램 흐름 중간에 3 점을 선택함으로써 사용자 좌표계 설정을 프로그램 중간에 임의로 변경 가능합니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면, 사용자 좌표계 설정은 기본 값으로 돌아옵니다.

Set 기능: XYZ Shift(목표 지점 위치 값 시프트)



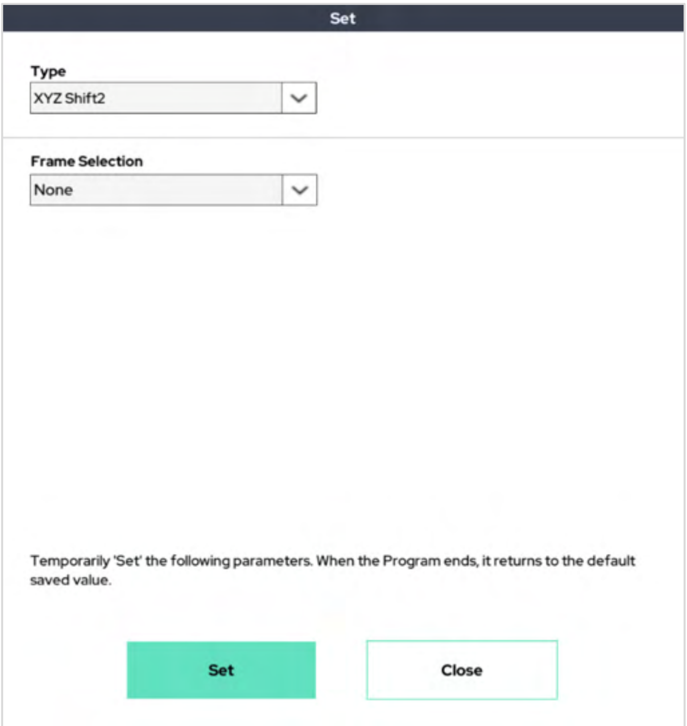
목표 지점을 임시로 바꿀 수 있는 기능입니다.

베이스/툴/사용자 좌표계를 선택할 수 있고 목표 지점으로부터 시프트 값을 입력합니다.

이때, 이 시프트를 L 계열 동작에만 적용할지, L 계열과 J 계열 동작 모두 적용할지 선택해줍니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

Set 기능: XYZ Shift2(목표 지점 위치 값 시프트 2)



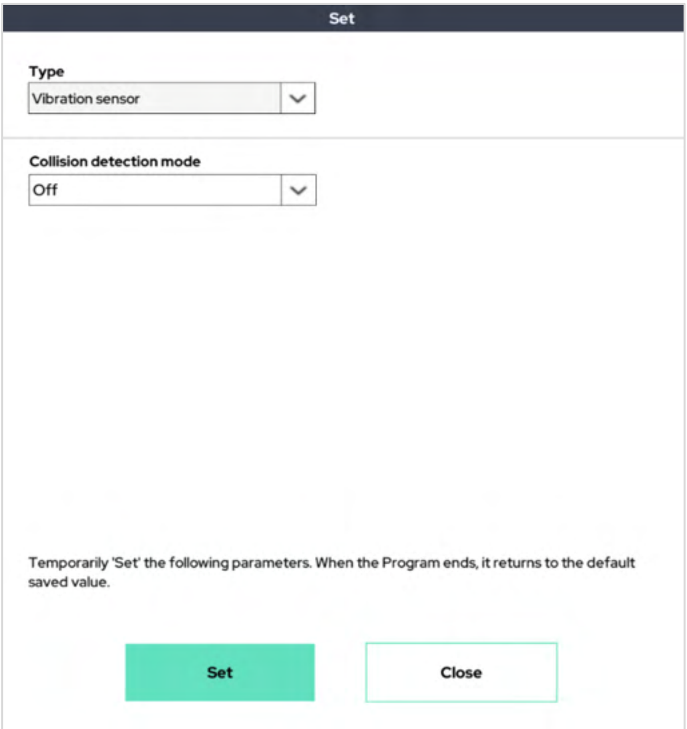
목표 지점을 임시로 바꿀 수 있는 기능입니다.

베이스/툴/사용자 좌표계를 선택할 수 있고 목표 지점으로부터 시프트 값을 입력합니다.

이때, 이 시프트는 L 계열 동작에만 적용되며, XYZ 위치 값과 회전 값 모두 시프트 값을 입력할 수 있습니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

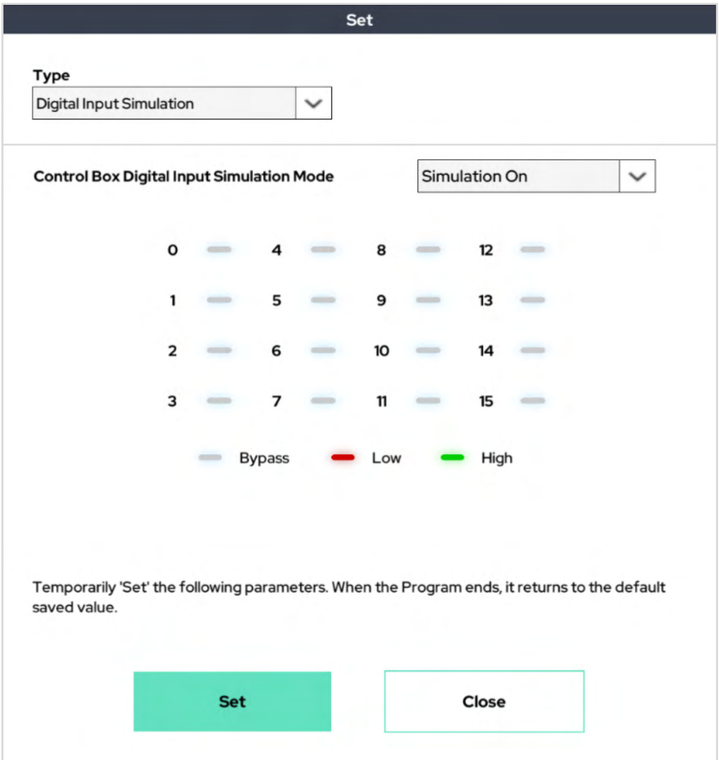
Set 기능: Vibration sensor(진동 센서 감지 On/Off)



충돌 감지 시 진동에 의한 충돌 감지는 제외하도록 하는 기능입니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

Set 기능: Digital Input Simulation(디지털 입력 신호 시뮬레이션)

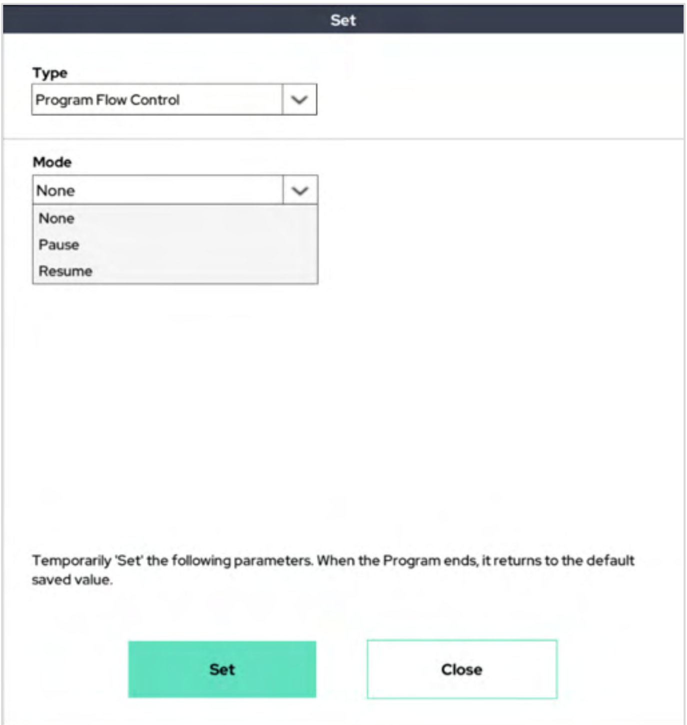


시뮬레이션을 통해 가상의 신호를 입력하는 기능입니다.

출력을 입력 받고자 하는 포트의 상태를 설정하여 원하는 입력을 만들어줍니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

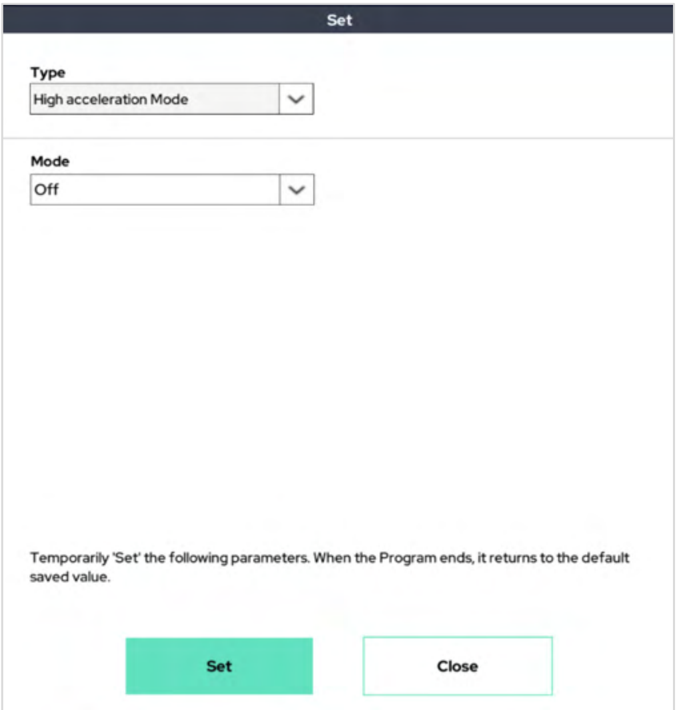
Set 기능: Program Flow Control(프로그램 흐름 제어)



프로그램이 실행되고 있을 경우 알람 및 I/O 를 사용하지 않고 일시 정지 및 재시작 할 수 있도록 하는 기능입니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

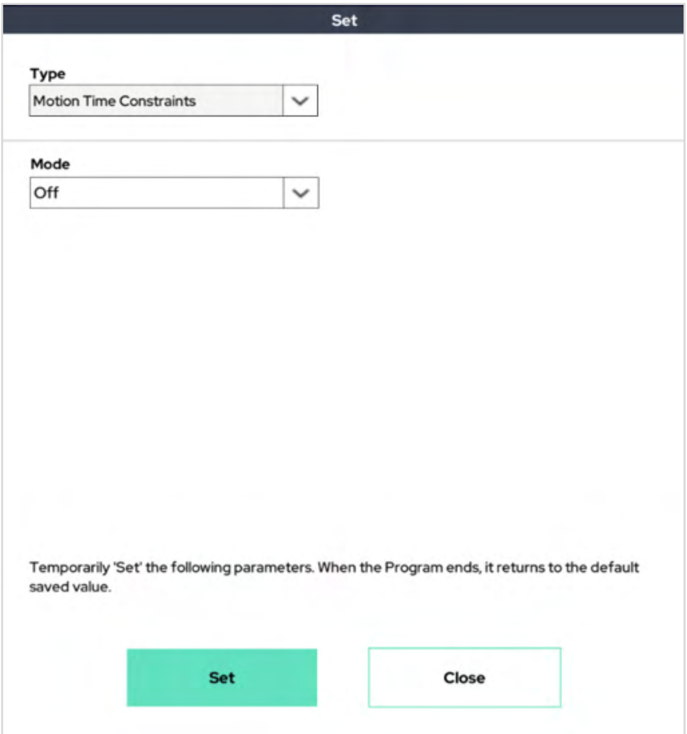
Set 기능: High acceleration Mode(고속 가속도 모드)



고속 가속도 모드는 감/가속 프로파일의 변화를 통해 로봇의 동작 속도가 원하는 동작 속도에 도달하는 시간을 줄여주는 기능입니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

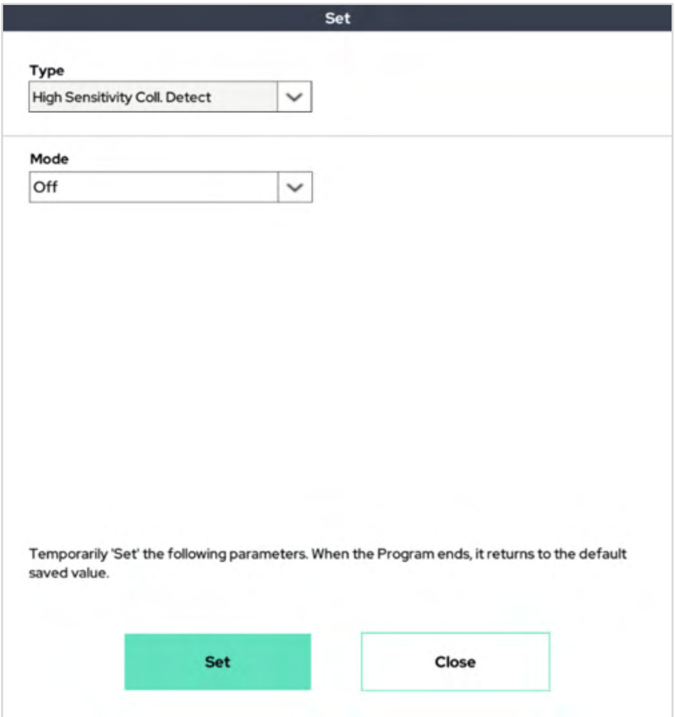
Set 기능: Motion Time Constraints(동작 시간 구속)



동작 시간 구속은 포인트에서 포인트로 이동 시에 걸리는 시간을 입력한 시간으로 구속하는 기능입니다. 이때 시간을 늘리는 것은 가능하지만 줄이는 것은 불가능합니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

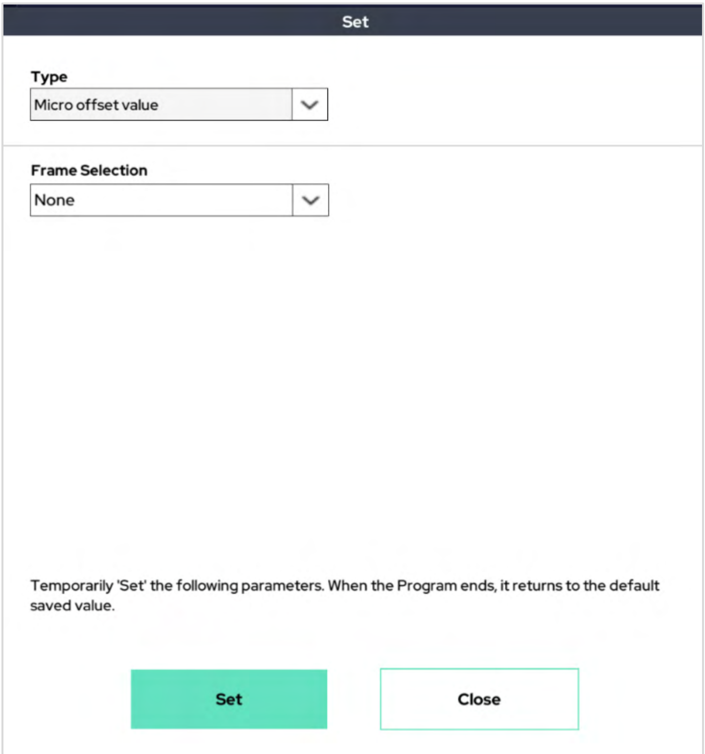
Set 기능: High Sensitivity Coll. Detect(고 민감도 충돌 감지)



고민감도 충돌 감지는 기존의 민감도보다 30% 더 민감하게 충돌의 감지가 가능하도록 한 기능입니다. Setup 에서 충돌감지를 가장 민감하게 한 둔감도 0%보다도 추가적으로 30% 민감해지게 됩니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

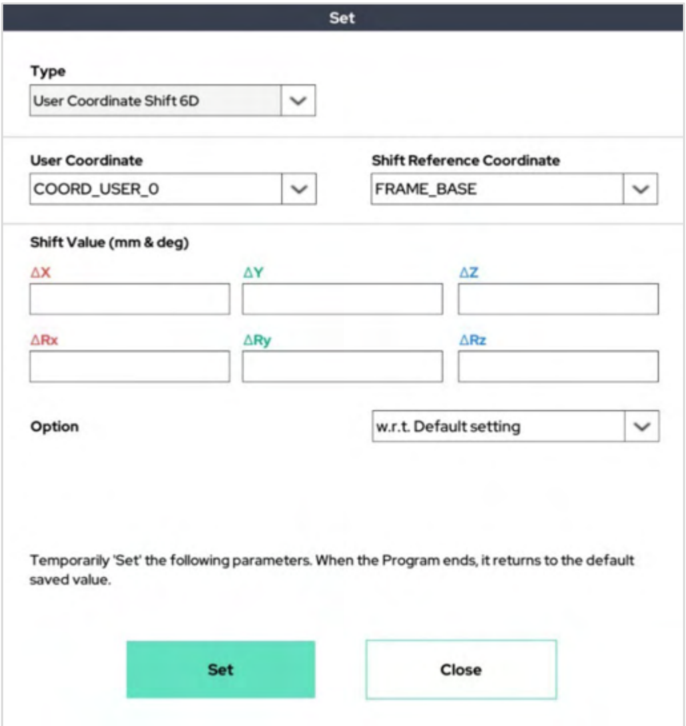
Set 기능: Micro offset value(오프셋 미세 조정)



원하는 좌표계를 기준으로 약간의 오프셋을 줄 수 있습니다. 이 기능을 통해 최대 20 mm 의 오프셋 임시 설정이 가능합니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

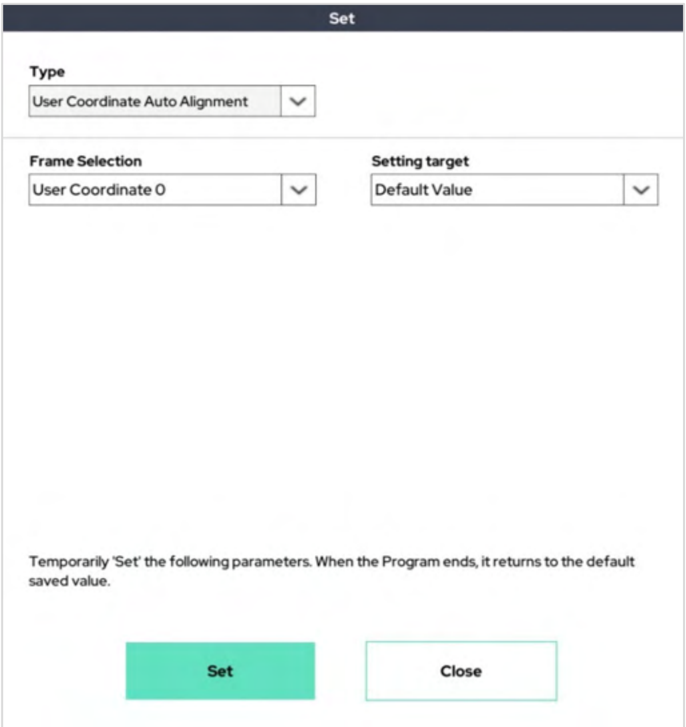
Set 기능: User Coordinate Shift 6D(사용자 좌표계 시프팅 6D)



사용자 좌표계를 임시로 시프팅 시킬 수 있습니다. 이 기능을 통해 사용자 좌표계의 위치, 회전을 임시로 변경할 수 있습니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

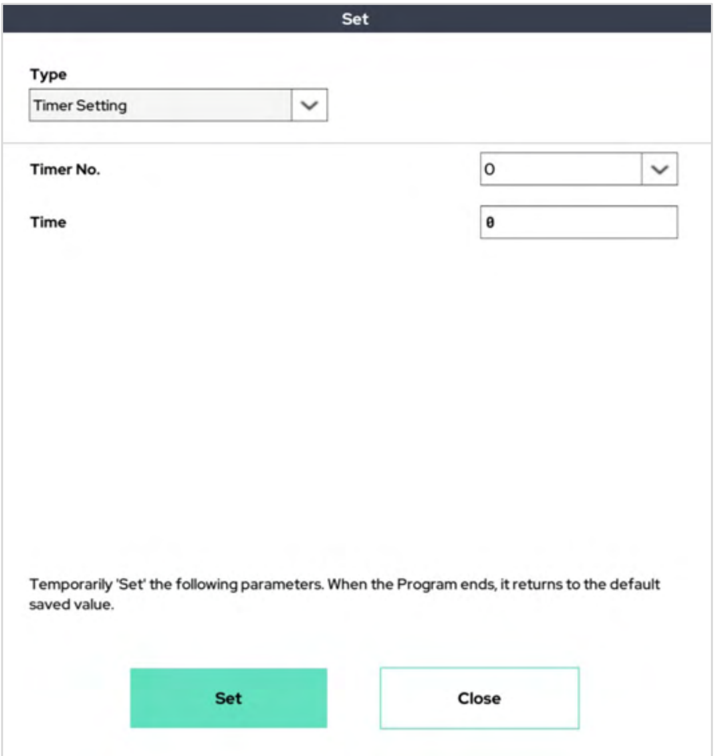
Set 기능: User Coordinate Auto Alignment(사용자 좌표계 자동 정렬)



선택한 사용자 좌표계를 기존 좌표계에서 현재 로봇의 TCP 좌표계로 변경시킬 수 있습니다. 다시 기존에 세팅해 놓은 사용자 좌표계로의 전환도 가능합니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

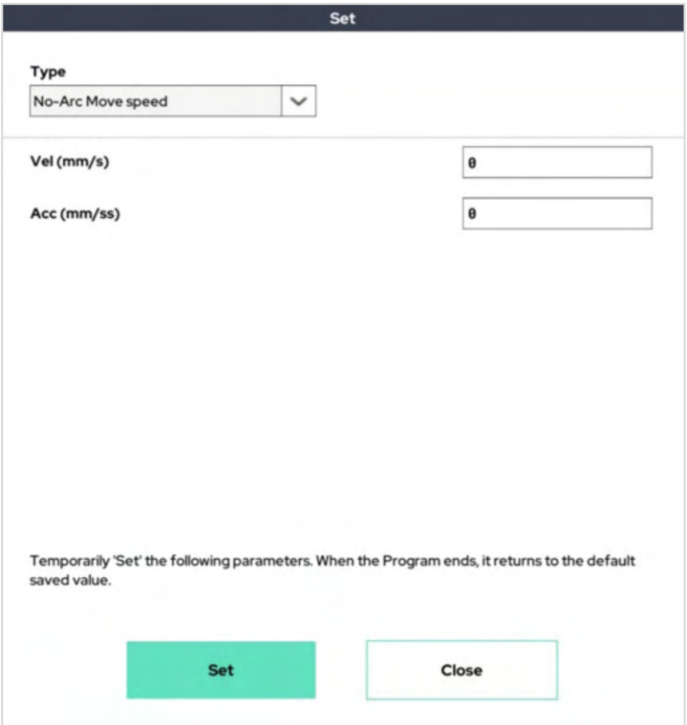
Set 기능: Timer Setting(타이머 세팅)



사용하고자 하는 타이머의 선택과 타이머의 초기 값을 세팅할 수 있습니다. 타이머는 세팅한 초기 값부터 시작합니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

Set 기능: No-Arc Move speed(노아크 동작 속도)



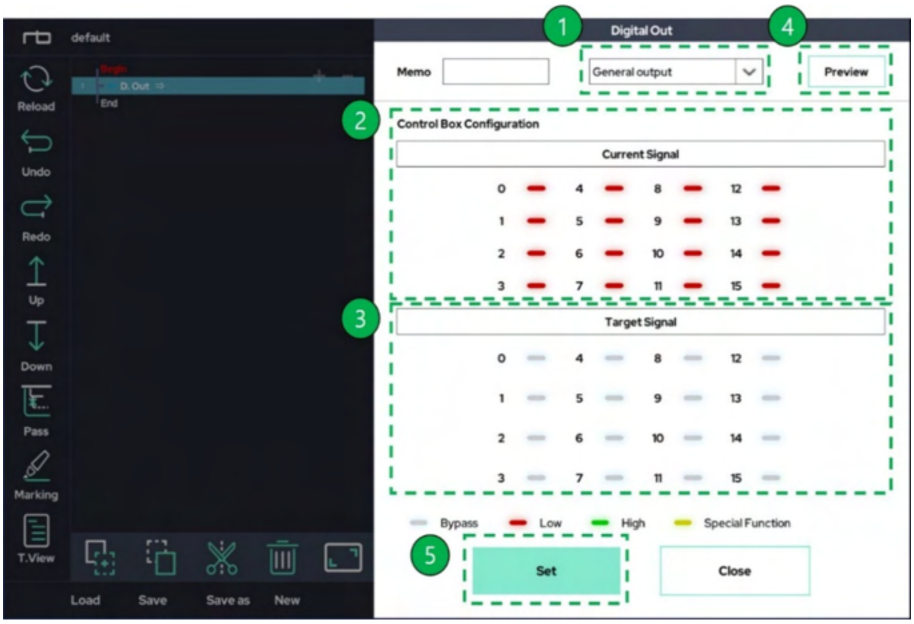
용접을 진행하지 않는 노아크 상태일 때 로봇의 동작 속도를 설정합니다.

Set 의 하위 기능이기 때문에, 프로그램이 끝나면 기본 값으로 돌아옵니다.

■ D.Out (Digital out) 기능 :

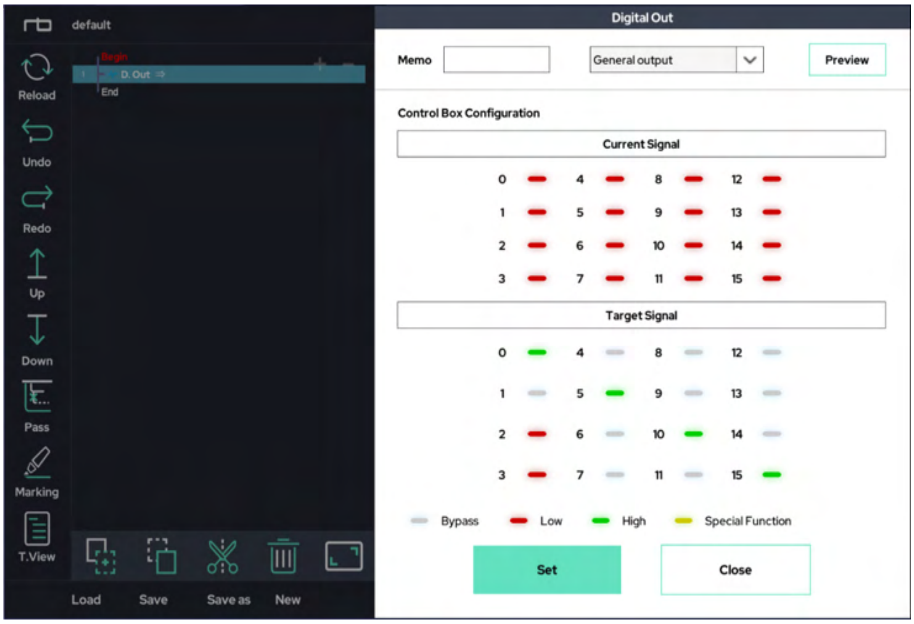
컨트롤 박스의 디지털 출력(Digital output)을 제어하는 명령어입니다. 16 개(0~15 번)의 포트 중 선택된 포트를 통해 선택된 신호가 출력되는 기능입니다. 각 포트에는 High 신호, Low 신호, Bypass 3 가지를 설정할 수 있습니다.

프로그램에 D.out 기능을 추가하고 클릭해보면 아래와 같은 팝업 창이 나타납니다.



① D.out 기능에서 사용할 수 있는 세부 기능을 선택합니다.
② 현재 컨트롤 박스에서 출력 중인 Digital Out 의 상태를 보여줍니다.
③ 원하는 설정은 선택하는 곳입니다. 3 단 토글 버튼을 통해 Bypass, Low, High 를 선택 가능합니다.
Bypass: 이전 출력 신호 상태를 유지합니다(회색).
Low: 출력 신호를 Low level 로 설정합니다(붉은색).
High: 출력 신호를 High level 로 설정합니다(초록색).
④ 3 번 항목에서 선택된 설정을 미리 보기로 출력해보는 기능입니다.
⑤ 3 번 항목에서 선택된 설정으로 D.out 기능을 설정합니다.

D.Out : 개별출력



컨트롤 박스와 태블릿 PC 가 연결된 상태에서, 목표 신호(Target Signal) 란을 위의 그림과 같이 설정 후, Preview(미리보기) 버튼을 누르면 아래와 같이 컨트롤 박스에서 Digital 출력 신호가 나갑니다.

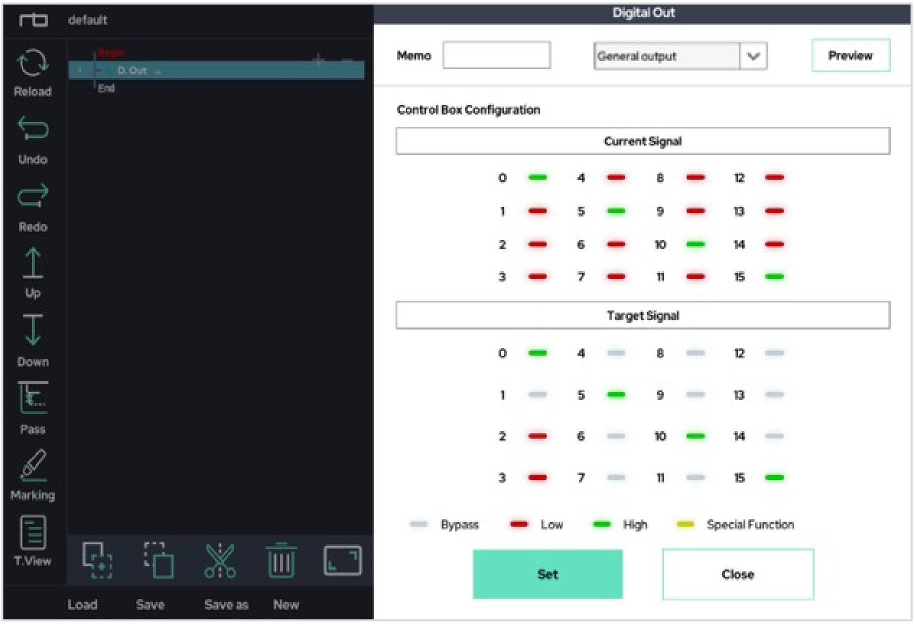
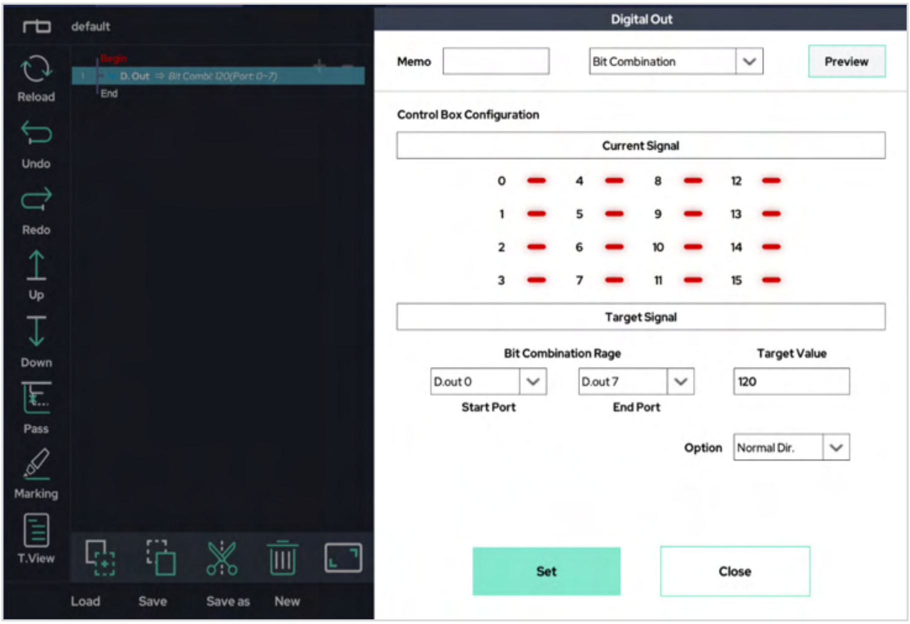


그림 우측의 Current Signal 란이 변경된 것을 볼 수 있습니다.

D.Out : 비트 조합 출력

사용하고자 시작 포트와 끝 포트를 선택한 후 원하는 값을 목표 값(Target Value)에 입력하면 비트 조합으로 Digital 출력을 내보낼 수 있습니다.



컨트롤 박스와 태블릿 PC 가 연결된 상태에서, 목표 신호(Target Signal) 란을 위의 그림과 같이 설정 후, Preview(미리보기) 버튼을 누르면 아래와 같이 컨트롤 박스에서 Digital 출력 신호가 나갑니다.

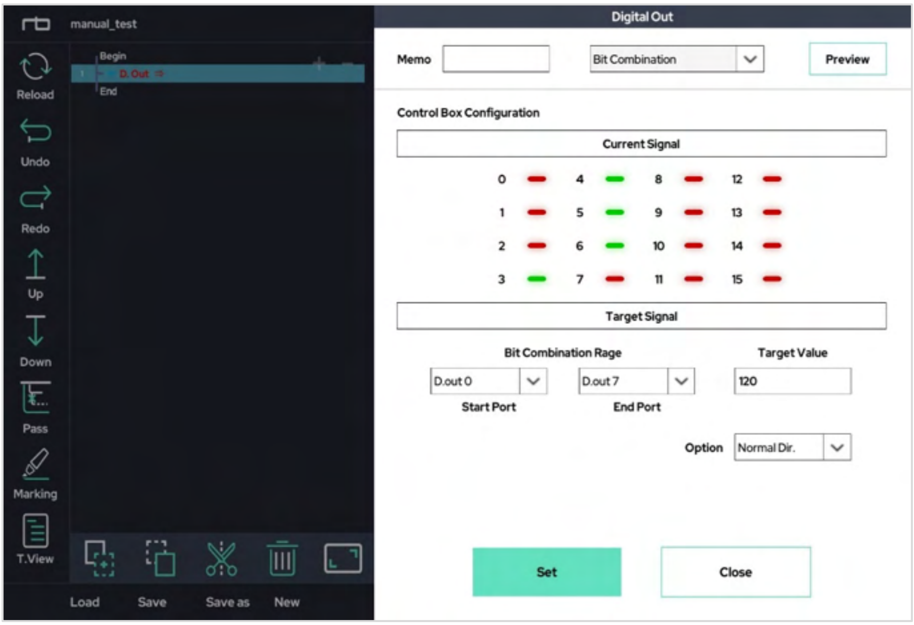
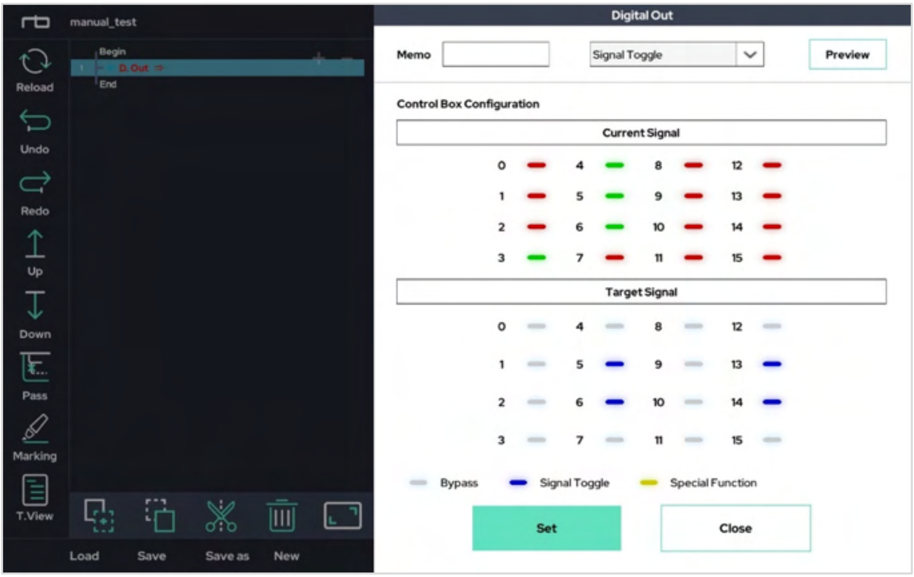


그림 우측의 Current Signal 란이 변경된 것을 볼 수 있습니다.

D.Out : 신호 토글 출력

High 인 신호는 Low 로 Low 인 신호는 High 로 출력 값을 바꾸어 신호를 출력합니다.



컨트롤 박스와 태블릿 PC 가 연결된 상태에서, 목표 신호(Target Signal) 란을 위의 그림과 같이 설정 후(토글 출력 신호는 파란색으로 표시), Preview(미리보기) 버튼을 누르면 아래와 같이 컨트롤 박스에서 Digital 출력 신호가 나갑니다.

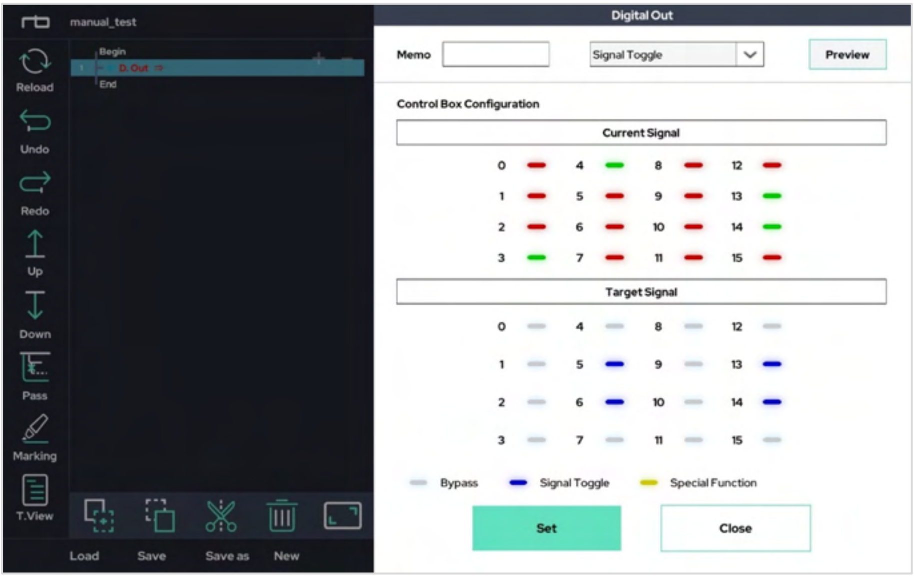
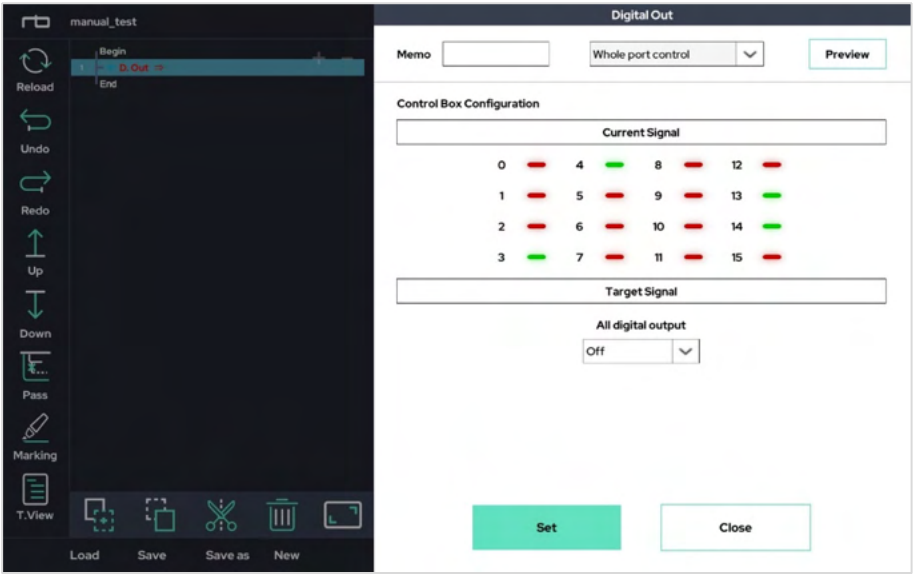


그림 우측의 Current Signal 란이 변경된 것을 볼 수 있습니다.

D.Out : 전체 신호 출력

전체 출력 포트의 신호를 한 번에 High/Low 로 전환할 수 있습니다.



컨트롤 박스와 태블릿 PC 가 연결된 상태에서, 목표 신호(Target Signal) 란을 위의 그림과 같이 설정 후, Preview(미리보기) 버튼을 누르면 아래와 같이 컨트롤 박스에서 Digital 출력 신호가 나갑니다.

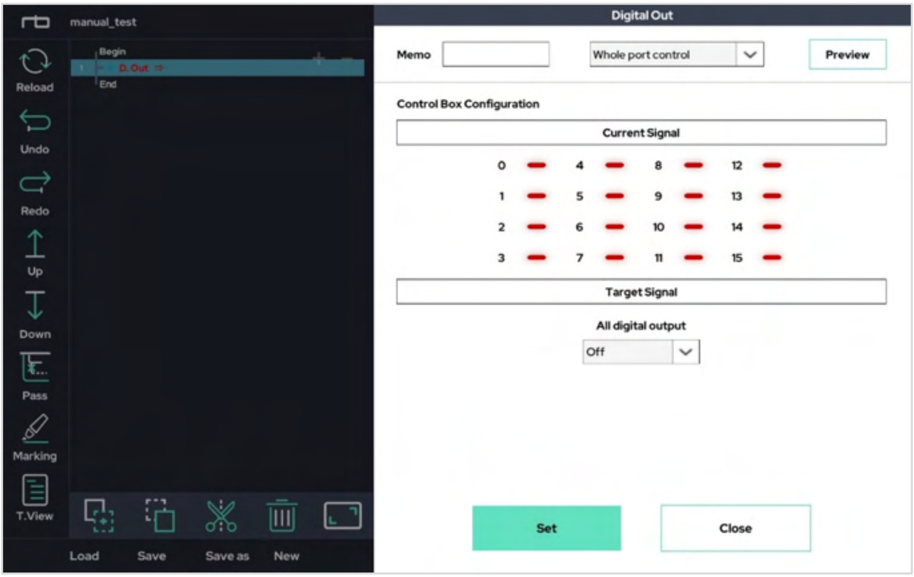
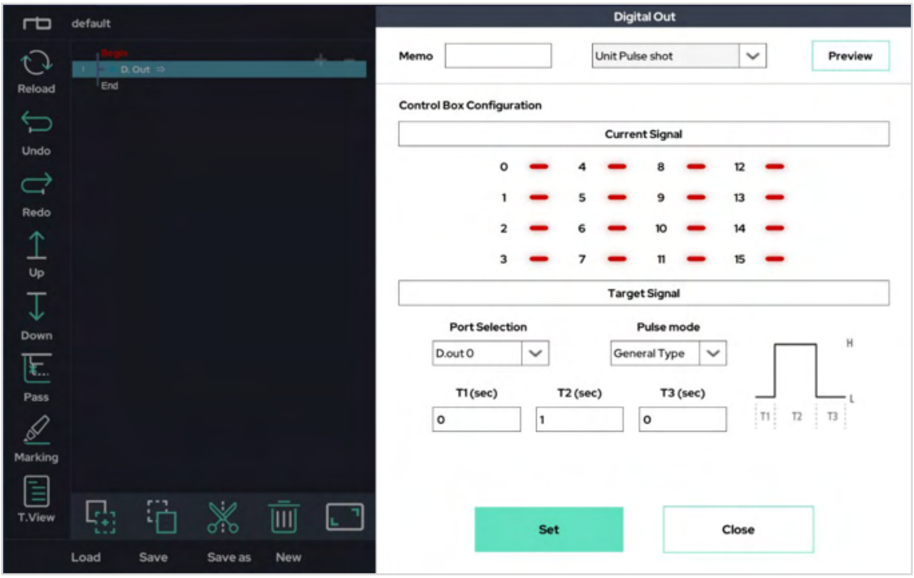


그림 우측의 Current Signal 란이 변경된 것을 볼 수 있습니다.

D.Out : 단위 펄스 출력

사용하고자 하는 포트를 선택하고 T1~T3 에 0~2 초 사이의 시간을 입력하면 입력 한 시간에 맞게 단위 펄스 신호를 출력합니다.



컨트롤 박스와 태블릿 PC 가 연결된 상태에서, 목표 신호(Target Signal) 란을 위의 그림과 같이 설정 후, Preview(미리보기) 버튼을 누르면 아래와 같이 컨트롤 박스에서 Digital 출력 신호가 나갑니다.

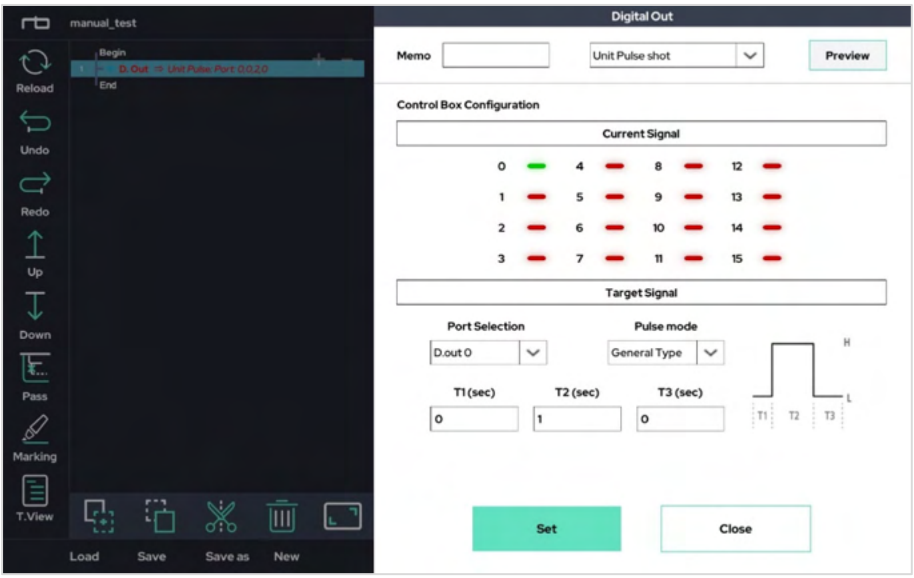
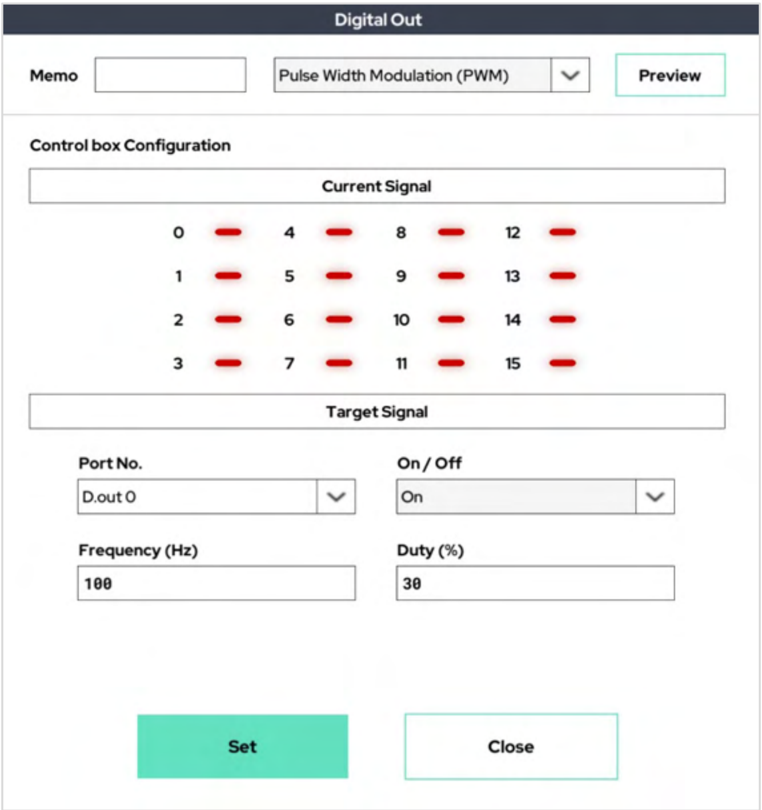


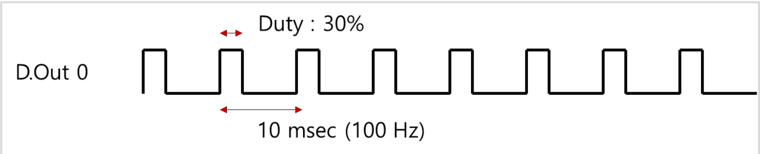
그림 우측의 Current Signal 란이 변경된 것을 볼 수 있습니다.

D.Out : 펄스 폭 변조(PWM)

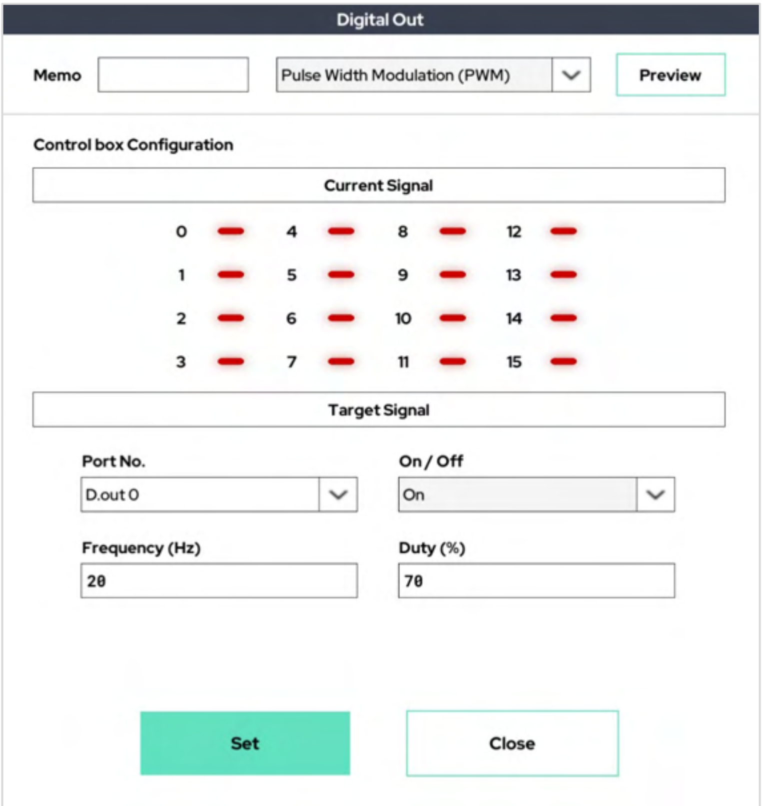
PWM (Pulse Width Modulation) 출력 기능입니다. PWM 펄스의 주파수와 듀티 비율을 입력하면 디지털 출력 포트를 통하여 PWM 신호를 송출하는 기능입니다.

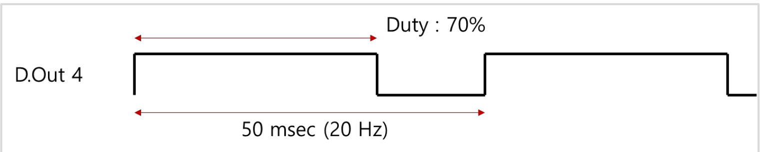
PWM 예시 1)





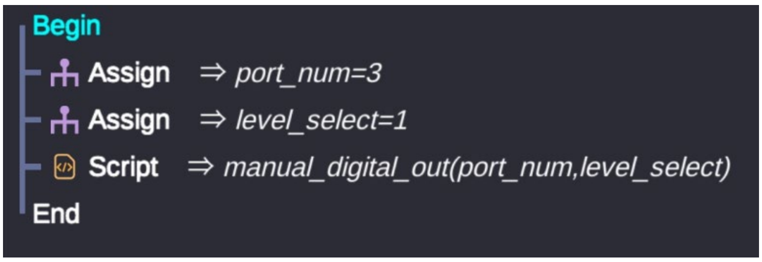
PWM 예시 2)





D.out 기능을 사용하는 것 이외에도 아래와 같이 Script 기능을 사용하여 Digital 출력을 내보내는 명령어를 만들 수 있습니다.

※ Script 함수: manual_digital_out(포트번호, 출력레벨)

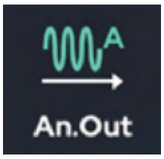


주의

주의:

1) 만약 Setup-I/O 에서 특정 Digital 출력 포트에 특수 기능을 할당해 두었다면, 해당 포트는 D.out 기능을 통해 사용하지 못합니다.
2) 특수 기능으로 할당된 포트는 D.out 창에서 노란색으로 표기됩니다.
3) D.out 창의 우측 상단에 Memo 기능을 활용하여, 해당 설정에 대한 메모를 남길 수 있습니다.
4) Digital 출력을 사용하기 전에 제조사가 제공하는 디지털 출력 포트의 전기적 특성을 충분히 숙지하고 사용하시기 바랍니다.

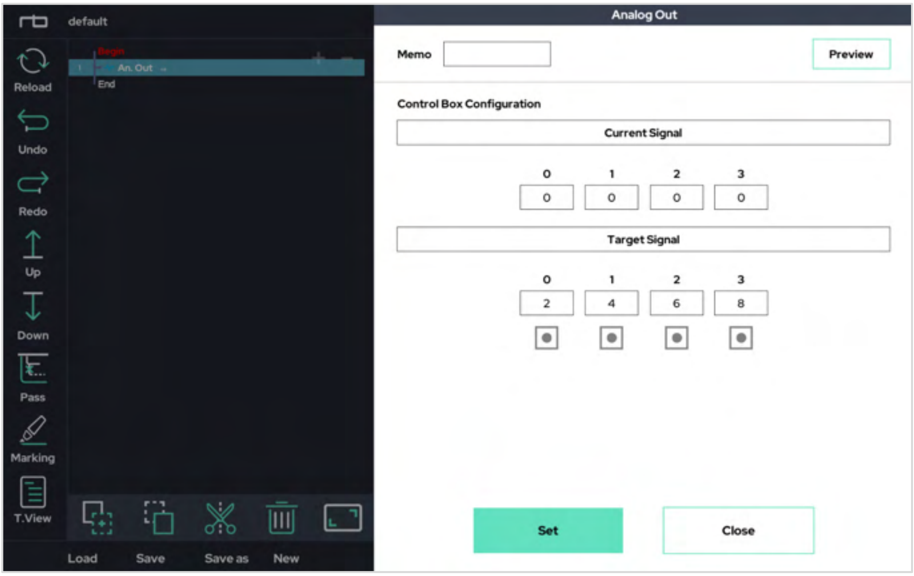
■ An.out(Analog out) 기능:



컨트롤 박스의 아날로그 출력 (Analog output)을 제어하는 명령어입니다. 4 개 (0~3 번)의 아날로그 포트 중 선택된 포트를 통해 선택된 전압을 출력하는 기능입니다. 각 포트는 0~10 V 의 전압 범주로 출력할 수 있습니다.

프로그램에 An.out 기능을 추가하고 클릭해보면 아래와 같은 팝업 창이 나타납니다.

① 현재 컨트롤 박스에서 출력 중인 Analog Out 의 상태를 보여줍니다. 원하는 전압 설정을 입력하는 란입니다. 체크 박스가 해제되어 있으면 Bypass 기능입니다.
② 기존의 전압 출력을 유지하는 설정입니다. 체크 박스에 체크를 하고, 원하는 전압을 0~10V 로 넣으면 전압 세팅이 됩니다.
③ 2 번 항목에서 선택된 설정을 미리 보기로 출력해보는 기능입니다.
④ 2 번 항목에서 선택된 설정으로 An.out 기능을 설정합니다.



컨트롤 박스와 태블릿 PC 가 연결된 상태에서, 목표 신호(Target Signal) 란을 위의 그림과 같이 설정 후, Preview(미리보기) 버튼을 누르면 아래와 같이 컨트롤 박스에서 Analog 출력 신호가 나갑니다.

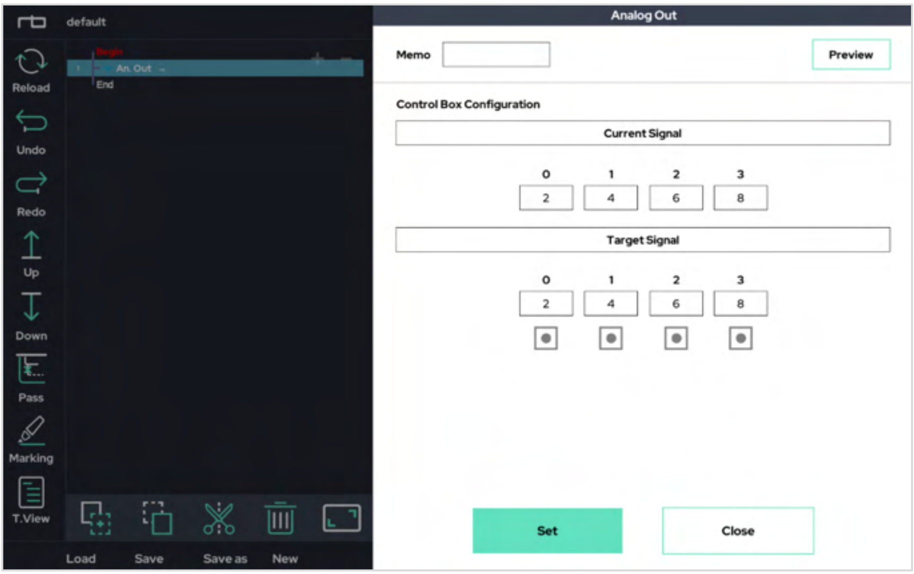
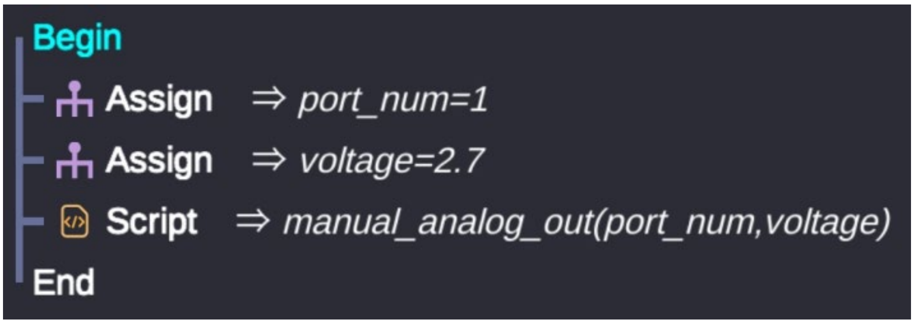


그림 좌측의 Current Signal 란이 변경된 것을 볼 수 있습니다.

An.out 기능을 사용하는 것 이외에도 아래와 같이 Script 기능을 사용하여 Analog 출력을 내보내는 명령어를 만들 수 있습니다.

※ **Script 함수**: manual_analog_out(포트번호, 출력전압)

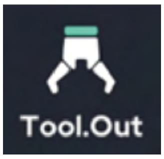


주의

주의:

1) An.out 창의 우측 상단에 Memo 기능을 활용하여, 해당 설정에 대한 메모를 남길 수 있습니다.
2) Analog 출력을 사용하기 전에 제조사가 제공하는 아날로그 출력 포트의 전기적 특성을 충분히 숙지하고 사용하시기 바랍니다.

■ Tool Out 기능:



툴 플랜지에는 2 개의 디지털 출력이 있습니다. 2 개의 디지털 출력의 신호를 지정할 수 있습니다. 더불어, 툴 플랜지에서 출력할 전압의 레벨(0V 또는 12V 또는 24V)을 함께 조절 가능합니다.

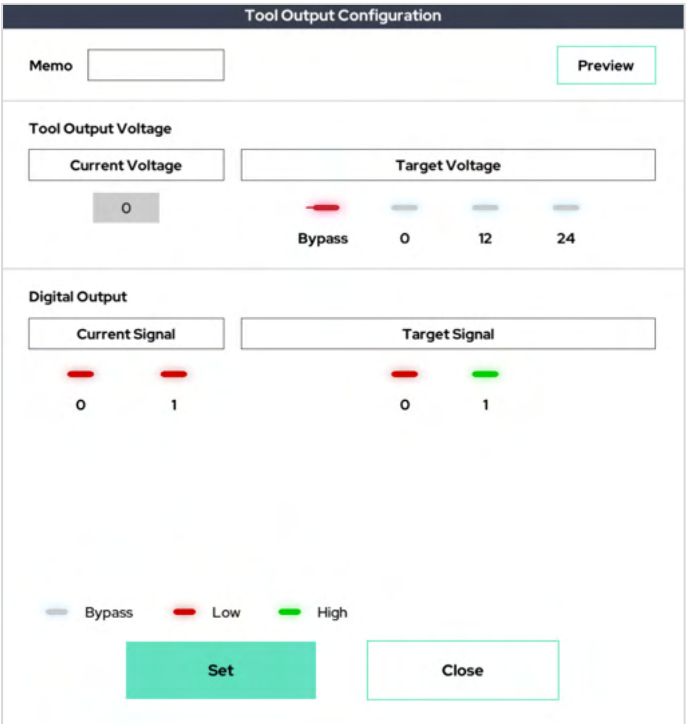
프로그램에 Tool 기능을 추가하고 클릭해보면 아래와 같은 팝업 창이 나타납니다.

① 현재 로봇 말단에서 출력 중인 툴플랜지의 상태를 보여줍니다.

② 원하는 출력 전압 및 digital 출력 설정을 하는 란입니다. 출력 전압은 Bypass, 0V, 12V, 24V 중에서 선택합니다. 디지털 출력은 3 단 토글 버튼으로 Bypass, Low, High 중 선택합니다.

③ 2 번 항목에서 선택된 설정을 미리 보기로 출력해보는 기능입니다.

④ 2 번 항목에서 선택된 설정으로 Tool 기능을 설정합니다.



컨트롤 박스와 태블릿 PC 가 연결된 상태에서, 로봇 활성화 이후, 목표 신호(Target Signal) 란을 위의 그림과 같이 설정 후, Preview(미리보기) 버튼을 누르면 아래와 같이 툴 플랜지 출력 신호가 나갑니다.

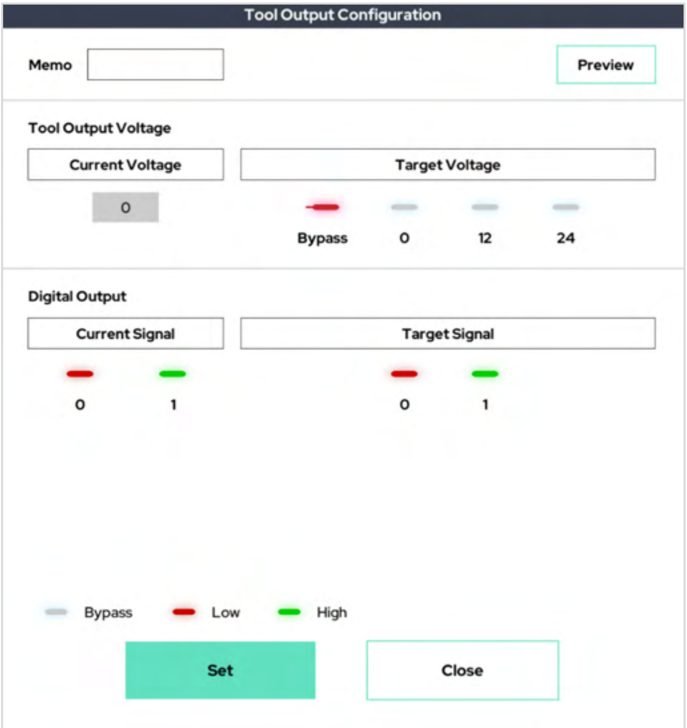


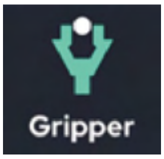
그림 좌측의 Current Signal 란이 변경된 것을 볼 수 있습니다.

7 주의

주의:

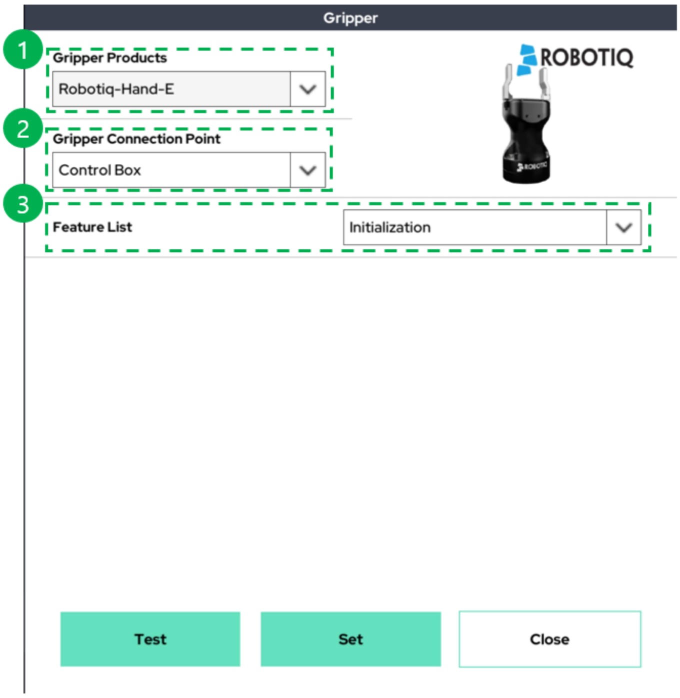
1) Tool.out 창의 우측 상단에 Memo 기능을 활용하여, 해당 설정에 대한 메모를 남길 수 있습니다.
2) 툴 플랜지 출력을 사용하기 전에 제조사가 제공하는 포트의 전기적 특성을 충분히 숙지하고 사용하시기 바랍니다.

■ Gripper 기능:



협동 로봇 전용 그리퍼를 위한 전용 기능입니다. Robotiq 사의 그리퍼 등 다양한 회사의 협동 로봇 그리퍼를 편리하게 테스트하고 사용 및 프로그램에 삽입 가능합니다. 단순 I/O 방식이 아닌, RS485 등의 시리얼 통신을 사용하거나 CRC 사용 등, 사용자가 프로그래밍해서 쓰기에 번거로운 그러퍼 사용을 편리하게 도와주는 기능입니다.

그리퍼 기능을 프로그램 Tree 에 추가합니다. 추가된 Gripper 를 클릭해보면 아래와 같습니다.



① 선택한 제조사에서 제공되는 그리퍼 제품군을 선택합니다.

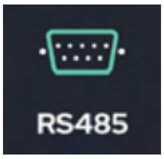
② 그리퍼가 연결되는 부분을 선택합니다(Control Box, Tool Flange).

③ 그리퍼로 사용할 기능을 선택합니다.

주의:

(1) 그리퍼 기능에서 제공되는 제품 목록은 사용자 피드백을 통해 업데이트 됩니다.

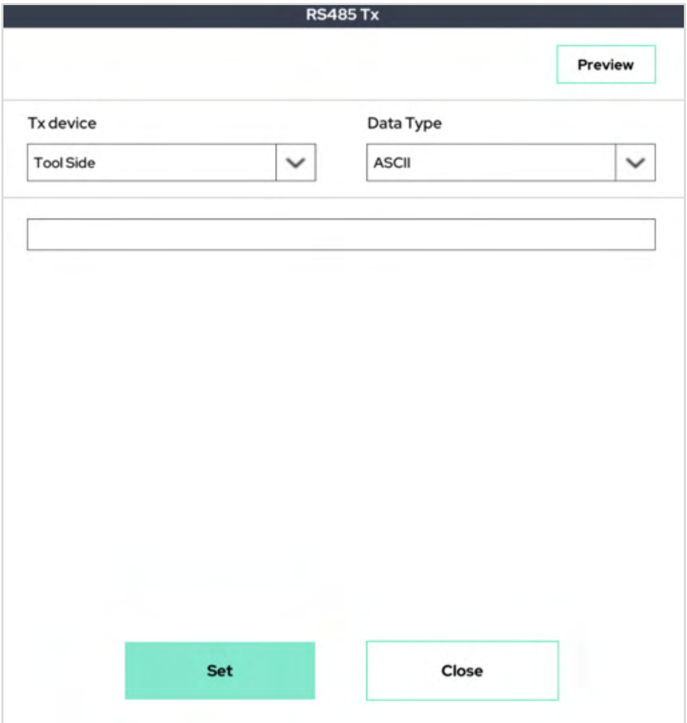
■ RS485 기능:



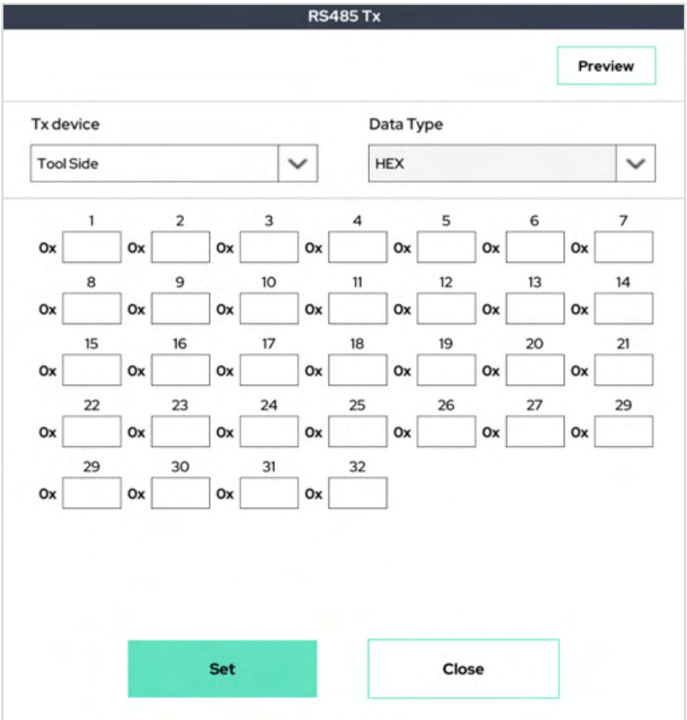
툴 플랜지를 통해, RS485 출력을 위한 기능입니다. 아스키(Ascii)모드로 출력을 줄 수도 있고, 16 진법(hex) 모드로 출력을 줄 수도 있습니다.

UI 태블릿을 통해서는 RS485 Tx 만 지원합니다.

구성된 통신은 팝업 창 우측의 Preview 버튼을 통해 미리 전송해 볼 수 있습니다.



[ASCII mode]

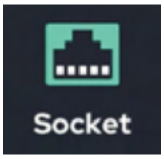


[HEX mode]

RS485 에서 사용할 통신 속도(Baud Rate), 기타 규약(Parity bit, Stop bit)은 Setup-serial 메뉴에서 설정합니다. 또는 프로젝트 상단에 Set-Serial_Configuration 옵션을 사용하여 설정 가능합니다.

박스 (Box side)에서 시리얼 통신을 사용하기 위해서는 상용 USB-Serial (RS232/422/485) 장치를 USB 포트에 꼽아 사용하면 됩니다.

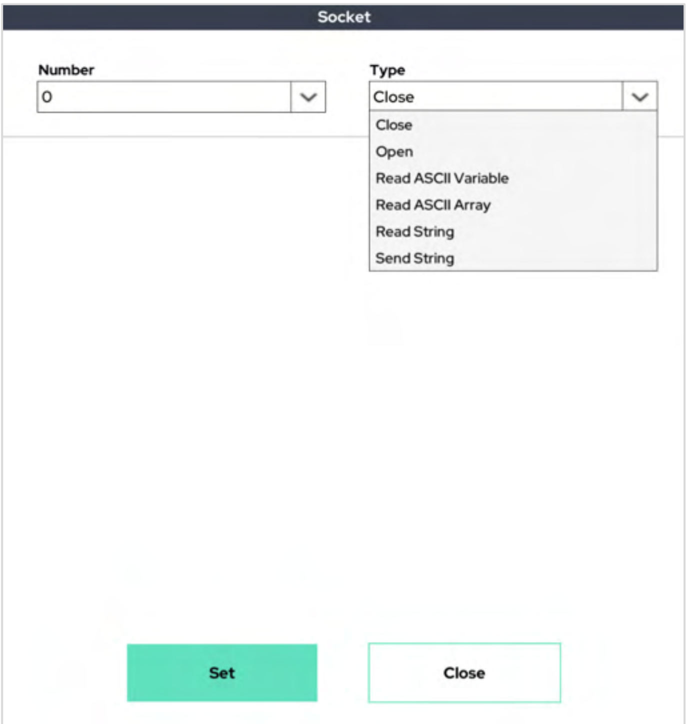
■ Socket 기능:



소켓 통신을 위한 기능입니다. 소켓을 열어서 특정 서버에 접속하고, 서버에 요청 메시지를 보내고, 서버로부터 특정 데이터를 받아오는 기능을 제공합니다. 소켓 통신은 최대 5 개까지 별도의 서버와 연결 가능합니다.

소켓 통신을 위한 컨트롤 박스의 IP 설정은 Setup 화면의 System 에서 설정 가능합니다.

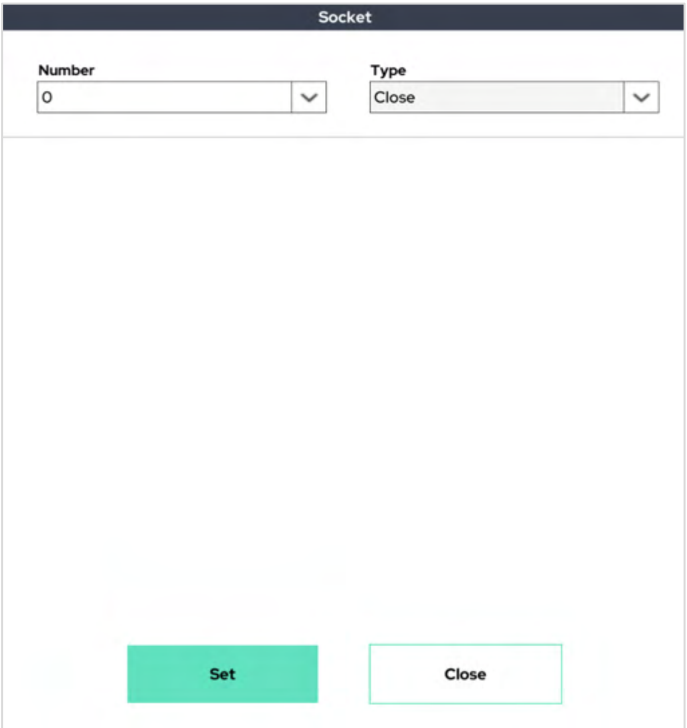
소켓 기능에서는 아래와 같이 6 가지의 기능이 제공됩니다.



- Close: 소켓을 닫습니다.
- Open: 소켓을 열고, 서버와 연결합니다.
- Read ASCII Variable: 서버로부터 온 문자열에서 숫자를 읽어 variable 타입 변수에 넣습니다.

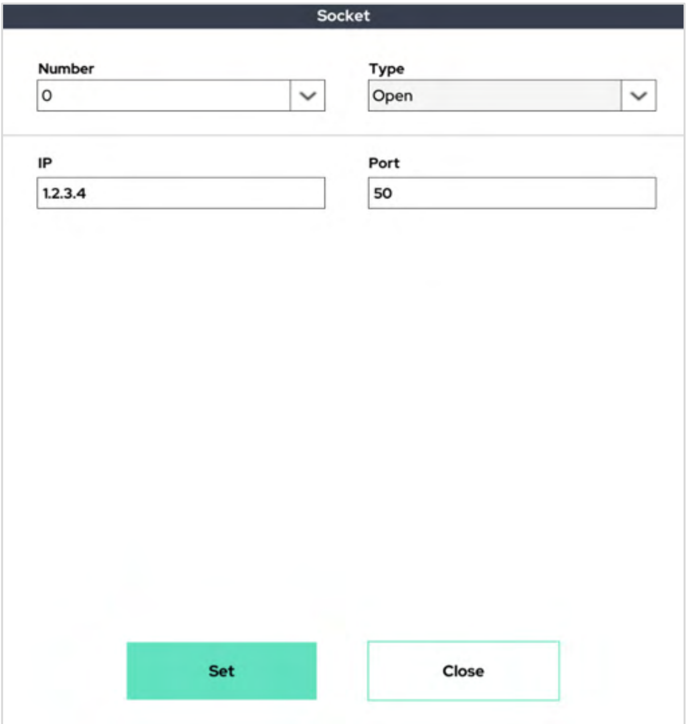
- Read ASCII Array: 서버로부터 온 문자열에서 배열을 읽어 array 타입 변수에 넣습니다.
- Read String: 서버로 온 문자열을 읽어 string 타입 변수에 넣습니다.
- Send String: 서버로 지정된 문자열을 보냅니다.

Socket 기능: Close



선택된 번호의 소켓 기능을 닫는 기능입니다.

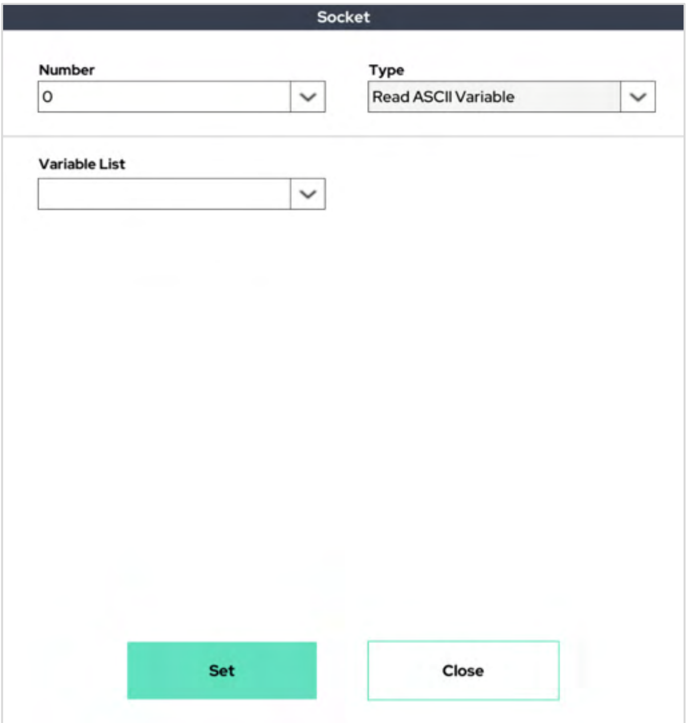
Socket 기능: Open



선택된 번호의 소켓 기능을 활성화하고, 상대 서버에 접속하는 기능입니다.

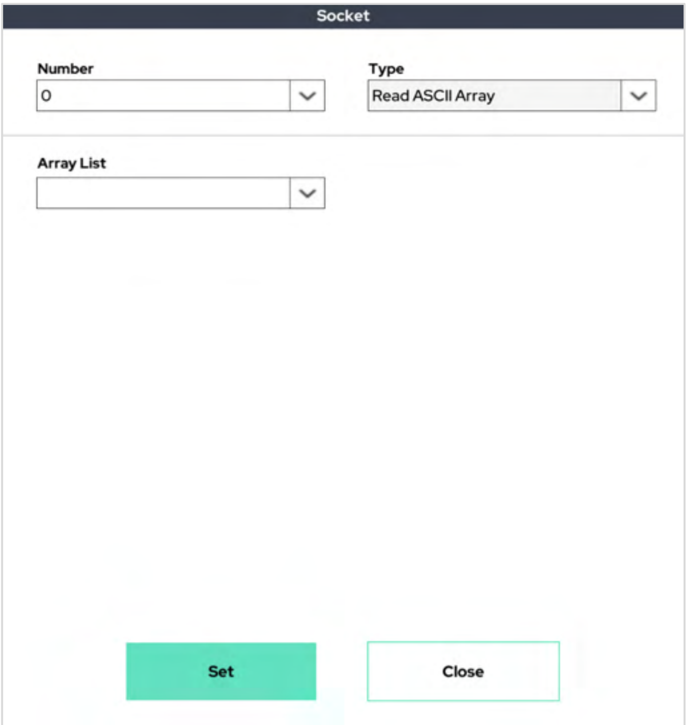
접속하고자 하는 상대 서버의 IP 주소와 포트 번호를 설정합니다.

Socket 기능: Read ASCII Variable



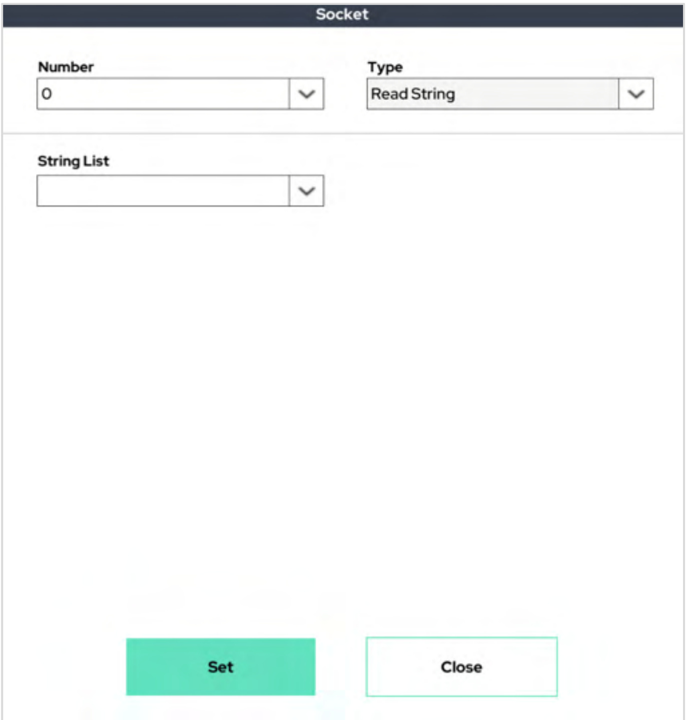
앞서 Assign 등을 통해 선언된 Variable 타입 변수 중 하나를 골라, 서버로부터 받은 문자열에서 숫자를 읽어서, 앞서 선언된 Variable 타입 변수에 넣는 기능입니다.

Socket 기능: Read ASCII Array



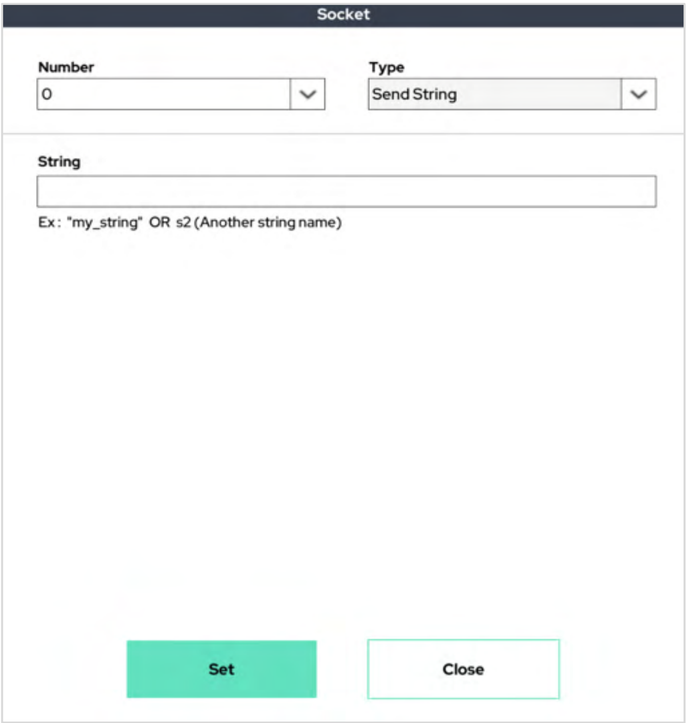
앞서 Assign 등을 통해 선언된 Array 타입 변수 중 하나를 골라, 서버로부터 받은 문자열에서 배열을 읽어서, 앞서 선언된 Array 타입 변수에 넣는 기능입니다.

Socket 기능: Read String



앞서 Assign 등을 통해 선언된 String 타입 변수 중 하나를 골라, 서버로부터 받은 문자열을 읽어서, 앞서 선언된 String 타입 변수에 넣는 기능입니다.

Socket 기능: Send String



서버로 특정 문자열을 보낼 때 사용하는 기능입니다. 직접 문자열을 입력하거나(예 “req_data”) 앞서 선언된 String 타입 변수를 보낼 수 있습니다.

주의

주의:

서버가 준수해야할 문법:

로봇 제조사에서 제공하는 Read ASCII Variable, Read ASCII Array, Read String 기능을 사용하기 위해서는 서버가 회신하는 데이터 형식은 아래와 같아야 합니다. 아래 문법 이외에 특수한 통신 문법이 요구되는 경우 제조사와 협의하세요.

Read ASCII Variable

서버는 숫자를 보낼 때, 숫자만을 작성하여 보내고자 하는 숫자를 보내야 합니다. (i.e. 따옴표 사이에 숫자를 적어 문자열로 숫자를 보내면 안 됩니다.)

(e.g. 123, 4567)

Read ASCII Array

서버는 배열을 보내고 싶을 때, 중괄호 {} 안에 숫자를 입력해서 보내야합니다. 숫자를 입력할 때, 숫자 사이에 쉼표(,)를 넣어 각 숫자를 구분해주어야 합니다. (i.e. 따옴표 사이에 배열을 적어 문자열로 배열을 보내면 안 됩니다.)

(e.g. {100,200,300})

Read String

서버는 "this_is_string_from_server"형식으로 따옴표를 포함한 문자열을 보내야 합니다.

소켓 통신을 돕기 위한 내부 변수:

소켓 통신의 연결 여부 / 회신 여부 등을 알기 위해 내부 변수가 아래와 같이 존재합니다. 아래의 두 내부 변수 모두 Shared Date 타입에서 선택 사용 가능합니다.

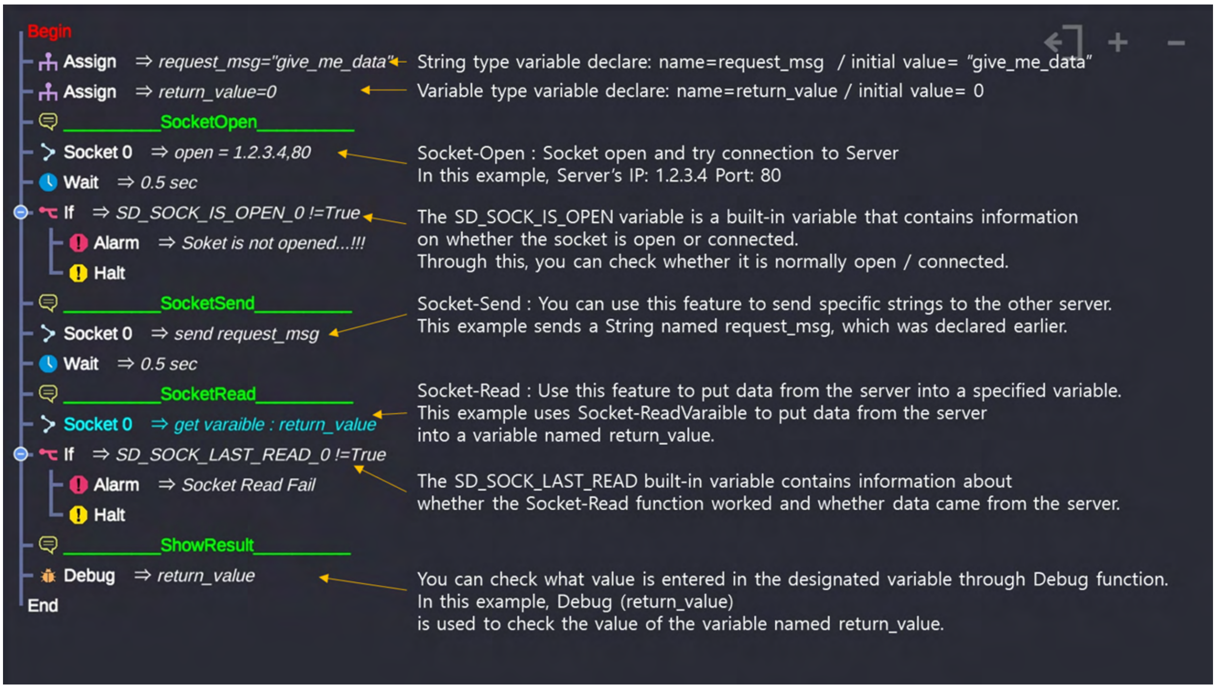
SD_SOCK_IS_OPEN_# (#은 소켓 번호 0~4)

해당 번호의 소켓이 정상적으로 Open 되었는지, 서버와 정상 연결되었는지 여부를 저장한 변수입니다. Socket-Open 이후에, If(SD_SOCK_IS_OPEN_0) 과 같은 식으로 소켓이 정상적으로 연결되었는지 체크할 수 있습니다.

SD_SOCK_LAST_READ_# (#은 소켓 번호 0~4)

해당 번호의 소켓으로 Read 기능이 정상적으로 수행되었는지 여부를 저장한 변수입니다. 예를 들어 Socket-ReadAsciiVariable 이후에, If(SD_SOCK_LAST_READ_0) 과 같은 식으로 마지막으로 수행한 Read 기능이 정상적으로 수행되었는지 체크할 수 있습니다. 서버로부터 들어온 데이터가 없을 경우 이 변수는 0 값을 가집니다.

아래 그림은 소켓 통신 예제를 보여줍니다.



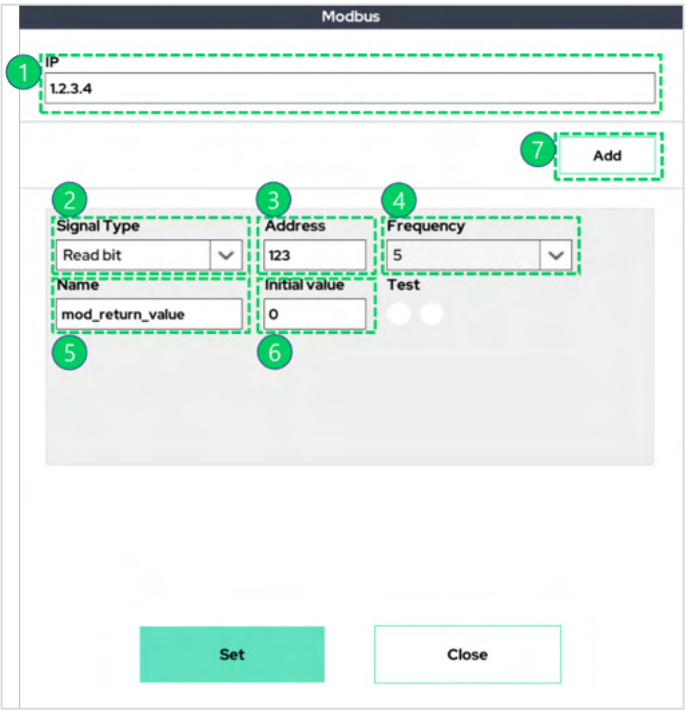
■ Modbus TCP(클라이언트) 기능:

모드 버스 클라이언트 기능입니다. 특정 IP/주소에 데이터를 요청하고, 반환된 데이터를 가져올 수 있는 기능을 제공합니다. 데이터 요청 주기와 형식 등을 지정 가능합니다.

모드 버스 TCP 의 포트 번호는 502 로 고정되어 있습니다 (Modbus 표준).

모드 버스 TCP 서버와 관련된 프로토콜 및 형식은 부록에 정리되어 있습니다.

모드 버스 TCP 클라이언트 기능은 프로그램 최 상단의 Pre.P 하위에 추가합니다.



① 모드 버스 클라이언트로 접속할 서버의 IP 주소 입력란

② 신호 타입을 선택 Read bit(1bit), Read word(16bit), Write bit(1bit), Write word(16bit) 중 하나를 선택합니다.

③ 서버에서 접근할 주소 값을 입력하는 란

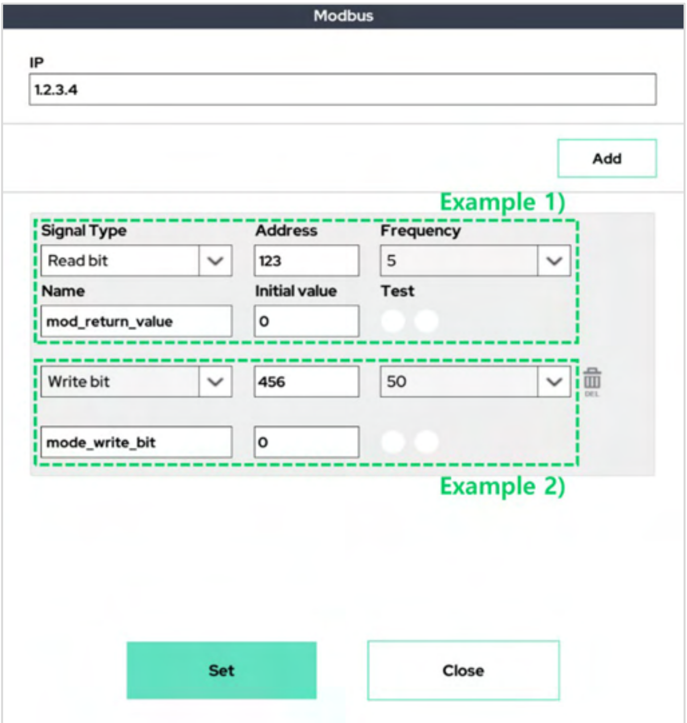
④ Read/Write 를 수행할 주기를 선택하는 란(Hz 단위)

⑤ Read 의 경우 읽은 값을 저장할 변수명을 선언합니다. Write 의 경우 출력할 변수명을 설정합니다.

⑥ 5 번에서 설정한 변수의 초기값

⑦ 사용할 신호를 추가하는 버튼

아래는 모드 버스 클라이언트 기능 설정에 대한 예시입니다.



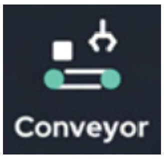
예시 1 해석)

서버 (IP: 1.2.3.4)의 주소 123 에 있는 word 타입 정보를 읽어서 mod_return_value 라는 변수에 넣는 기능. 정보를 읽는 주기는 5 Hz.

예시 2 해석)

서버 (IP: 1.2.3.4)의 주소 456 에 mod_write_bit 에 저장된 값을 쓰는 기능. 50 Hz 의 주기로 mod_write_bit 의 값을 서버로 전송.

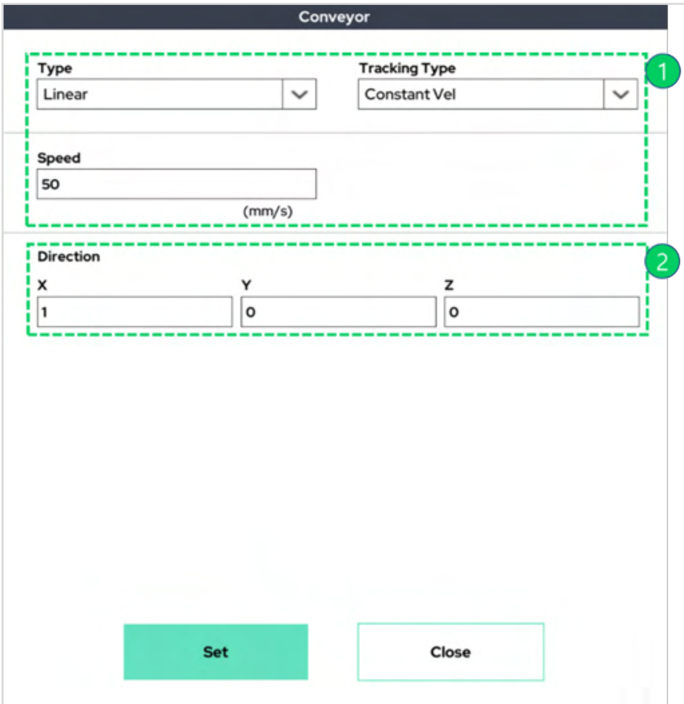
■ Conveyor 기능:



컨베이어 추적 기능입니다. 컨베이어 속도와 방향에 관한 설정을 하면, 컨베이어 기능의 하위 동작 (MoveL, MoveLB, Circle 등)은 컨베이어 흐름에 더해집니다.

MoveJ, MoveJB 등 Joint 동작 계열의 동작은 Conveyor 의 하위 동작에 넣지 못합니다. MoveL, MoveLB, MovePB, MoveJL, MoveITPL, Circle 만 지원됩니다.

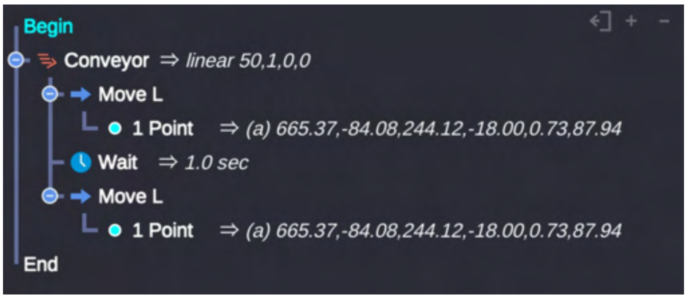
컨베이어 기능을 프로그램 Tree 에 추가하고, 이 기능을 클릭하여 옵션을 보면 아래와 같습니다.



① 컨베이어의 형태와 속도, 트래킹 대상 등을 설정합니다.

② 컨베이어 진행 방향에 대한 벡터 입니다(x,y,z 값은 로봇팔 베이스 좌표계 기준입니다).

이를 바탕으로 프로그램을 구성하면 아래와 같습니다.



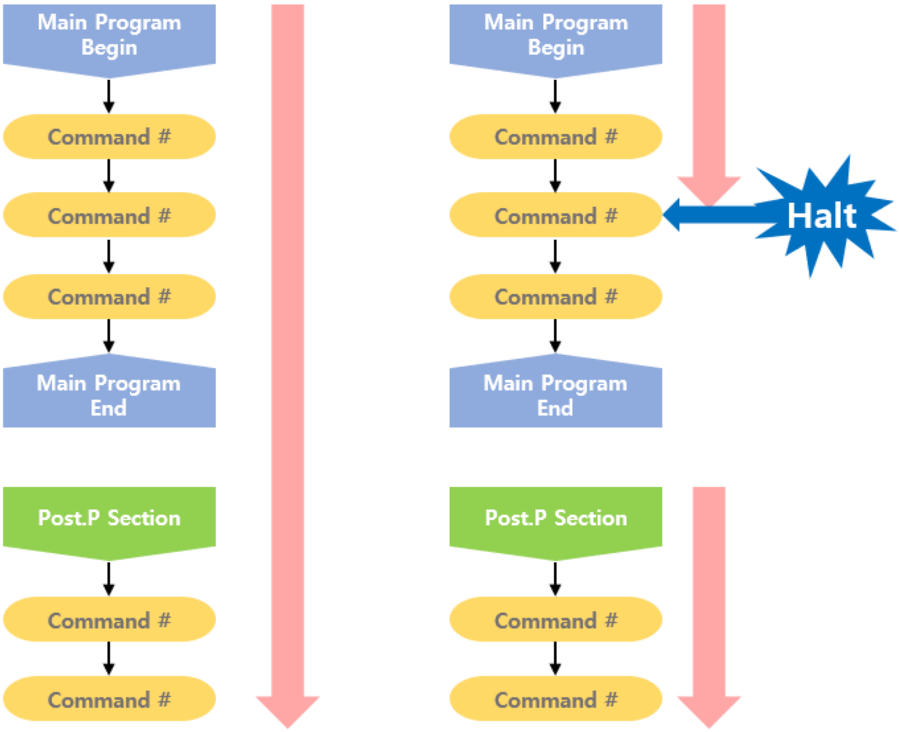
■ Post.P(Post Program) 기능:

프로그램 종료 후 마무리 작업으로 수행할 기능을 정의합니다.

Post Program 기능은 프로그램 종료 후 실행되는 명령어를 넣는 기능입니다.

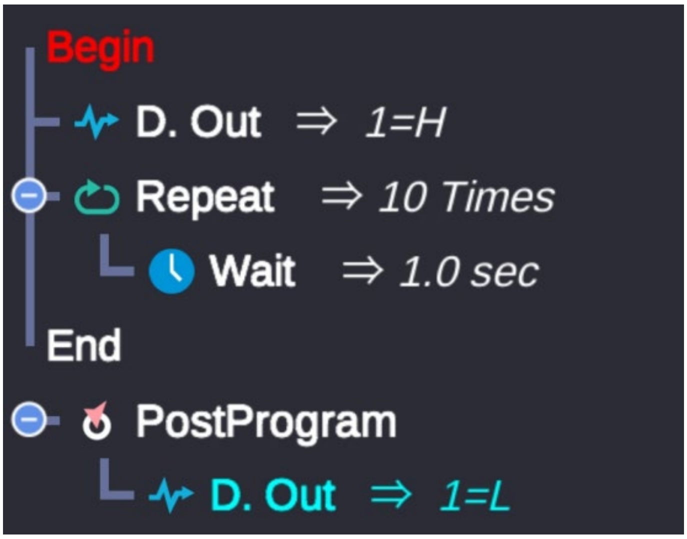
Post.P 하위에 선언된 명령어는 프로그램 종료 후 순차적으로 수행됩니다.

Post.P 의 실행은 아래 다이어그램과 같이 진행됩니다.



Post.P 예시 1)

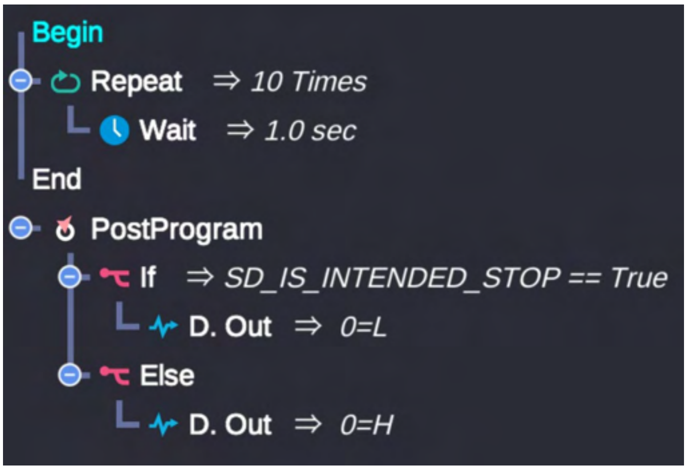
아래의 예시는 Post.P 를 사용한 예시입니다. 프로그램 시작과 동시에 (Begin 하단에) D.out 기능을 통하여 1 번 포트에 High 신호를 송출합니다. 프로그램 종료 전에 (End 이전에) D.out 1 번을 끄는 것을 넣지 않았지만, 프로그램이 종료되면 1 번 포트는 자동으로 Low 신호를 송출하게 됩니다. 이는 Post.P 기능에 D.out 1=L 를 사용했기 때문입니다.



위의 예시와 같이, Post.P 기능은 안전 기능 등에 활용할 수 있습니다.

Post.P 예시 2)

아래의 예시는 Post.P 를 사용한 예시입니다. 프로그램이 정상 종료되면 경광등 (D.out 0 번에 연결)을 켜지 않고, 비정상 종료되면 경광 등을 키는 예시입니다.



본 예시에서 사용된 SD_IS_INTENDED_STOP 은 시스템 내부 변수로써, 프로그램 시작 시 항상 0 (거짓, false)으로 초기화 됩니다. 사용자 의도에 의해 정상적으로 프로그램이 멈출 경우 이 변수는 1 (참, true)이 됩니다. 만약, 여러 이유에 의해서 프로그램이 비정상으로 멈출 경우 이 변수는 거짓으로 남아 있습니다.

UI 종료 버튼 클릭에 의한 종료, I/O Stop 신호에 의한 종료, 기타 통신을 통한 종료 신호, 등 사용자 선택에 의한 종료 신호는 정상 종료로 판별합니다(SD_IS_INTENDED_STOP = 참).

만약 특이점 접근으로 인한 프로그램 종료, 명령어 문법 문제로 인한 종료 등, 사용자 의도가 아닌 종료는 비정상 종료로 판별합니다(SD_IS_INTENDED_STOP = 거짓).

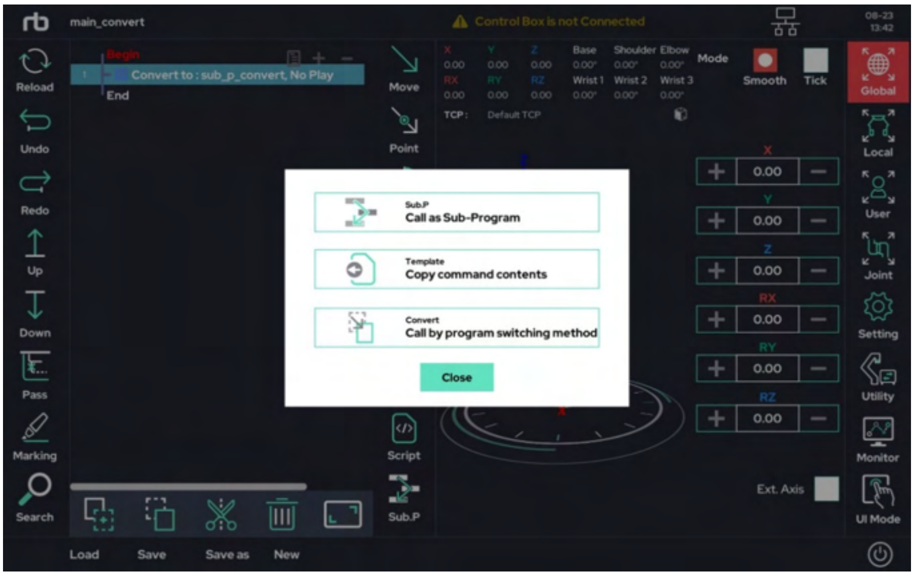
- 프로그램이 정상 종료되지 않은 경우(예: Alarm 에서 Halt 정지를 누른 경우)에도 Post Program 에 정의된 기능은 수행됩니다.
- MoveJ, MoveL 등 로봇 팔의 동작에 관한 명령어는 Post.P 의 하위에서 사용하지 못합니다.
- Post.P 기능은 최상위 프로그램에서만 작동합니다. Sub.P 기능을 통하여 호출된 하위 프로그램에서 Post.P 기능을 사용할 경우 해당 부분은 실행되지 않습니다.

■ Template 기능:

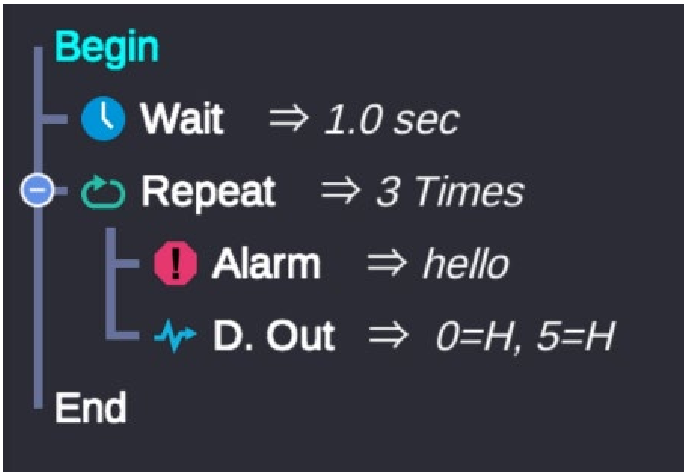
사전에 만들어진 다른 프로그램 파일(티칭 파일)을 현재 문서에 수정 가능한 형태로 삽입하는 기능입니다.

Sub.P 의 기능과 유사하지만, Sub.P 의 경우 단순히 해당 프로그램을 호출하는 기능이라면, Template 기능은 현재 프로그램에서 수정 가능한 형태로 다른 프로그램을 복사해서 들고 오는 기능입니다.

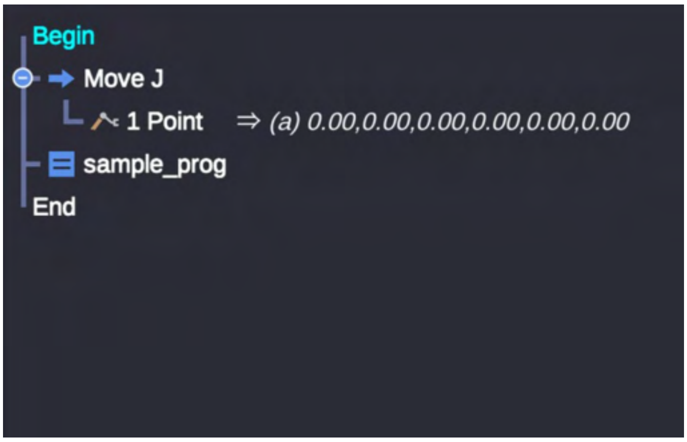
프로그램에서 Sub.P 아이콘을 클릭하면 아래와 같은 팝업창이 나오고 이때, 명령어 내용 복사를 누릅니다.



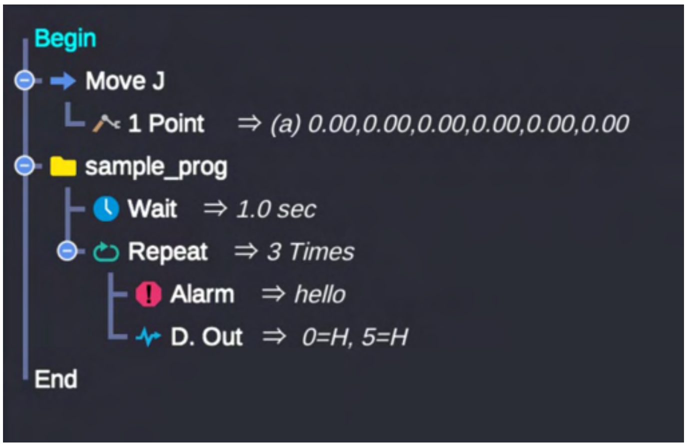
아래와 같이 “sample_prog”라는 이름의 프로젝트가 만들어졌다고 가정합니다.



예시 1) sample_prog 를 Sub.P 기능을 사용하여 불러올 경우



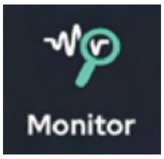
예시 2) sample_prog 를 Template 기능을 사용하여 불러올 경우



예시 1)에 나타난 것처럼 Sub.P 로 불러올 경우, 해당 프로젝트를 수행은 하지만, 상위 프로그램에서 수정이 불가능 형태가 됩니다. 또한, 불러온 하위 프로젝트가 변경되게 되면 상위 프로그램의 구동도 바뀌게 됩니다.

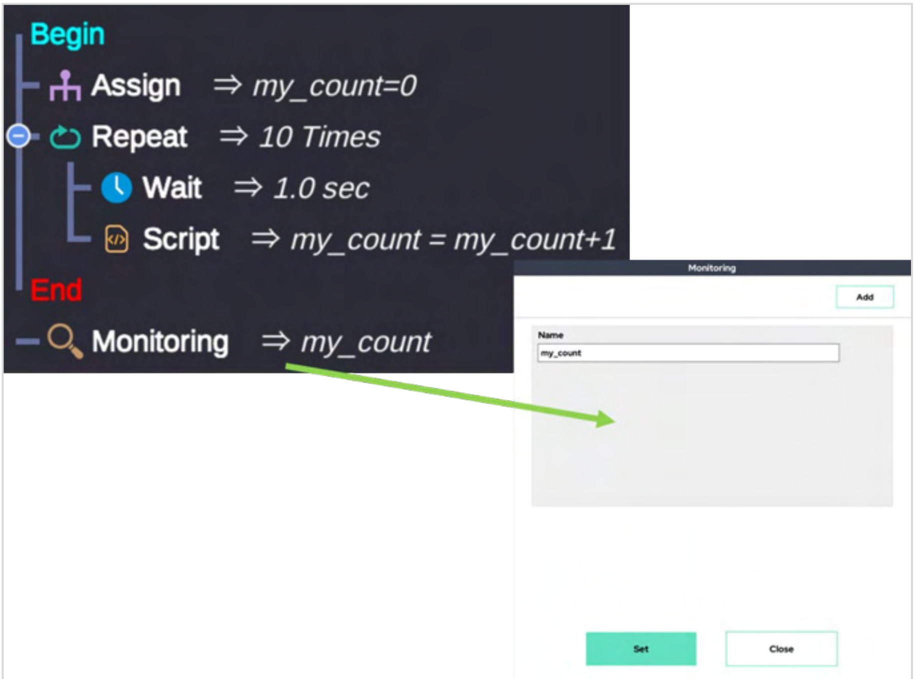
예시 2)와 같이 Template 기능으로 불러온 경우, 상위 프로그램에서 수정이 가능한 형태로 불러와 집니다. 한번 Template 으로 복사되어 들어온 내용은, 원본 하위 프로그램이 수정되어도 바뀌지 않습니다.

■ Monitor 기능:



프로그램 구동 중, 실시간으로 관찰하고 싶은 변수 (단일 변수, 배열, 포인트 변수 등)를 선언하는 기능입니다.

Monitor 기능에 선언된 변수들은 Make/Play 페이지 우측의 모니터 아이콘을 클릭하면, 프로그램 구동 중 실시간으로 변수 값을 관찰 가능합니다.



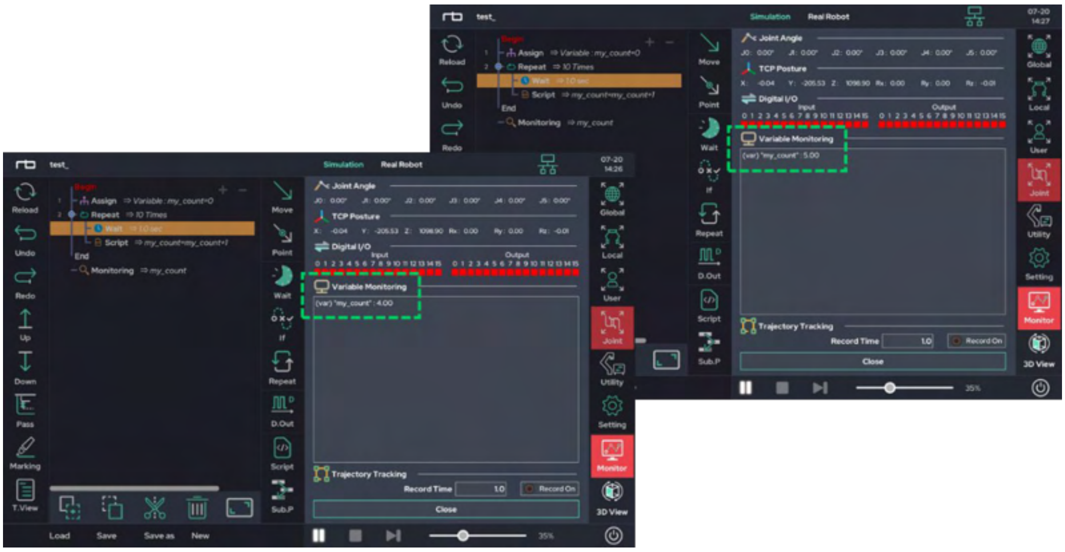
위의 프로그램 예시에서는 my_count 라는 이름의 변수를 하나 선언하고, 반복문 (Repeat)을 통하여 1 초에 1 씩 my_count 를 증가시키는 기능을 구현하였습니다.

그리고, Monitor 기능을 사용하여 관찰할 대상으로 my_count 를 지정하였습니다.

Monitor 기능에는 관찰하고자 하는 변수의 이름을 넣으면 됩니다.

프로그램 구동 중, Monitor 에 지정한 변수의 값을 실시간으로 관찰하고 싶을 경우, 화면 우측의 Monitor 아이콘을 클릭하면 됩니다.

이후, 프로그램을 재생 (Run, ▷)하면, 1 초마다 my_count 의 값이 증가되는 모습을 실시간으로 관찰할 수 있습니다.



■ Pattern 기능:

반복 동작을 정의하는 기능입니다.

반복 동작을 수행할 공간에 대한 정보를 정의하고, 각 위치에서 수행할 반복 동작을 정의하면, 모든 공간 지점에서 같은 동작을 수행합니다.

이 기능을 통하여 팔래타이징(palletizing)을 구현 가능합니다.

3 가지의 하위 설정이 있습니다.

- Pattern Property:

  반복 동작을 수행할 대상 면/공간을 정의합니다.

  직선, 평면, 3 차원큐브, 임의의 지점 등 다양한 형태를 지원합니다.

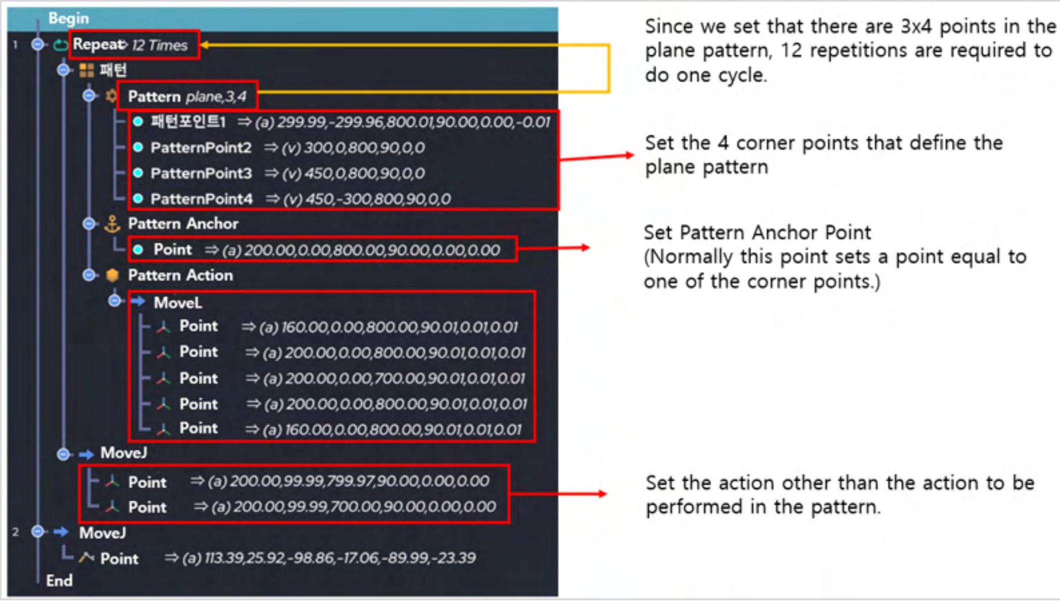
- Pattern Anchor:

  Pattern Action 에서 정의할 동작의 기준점입니다.

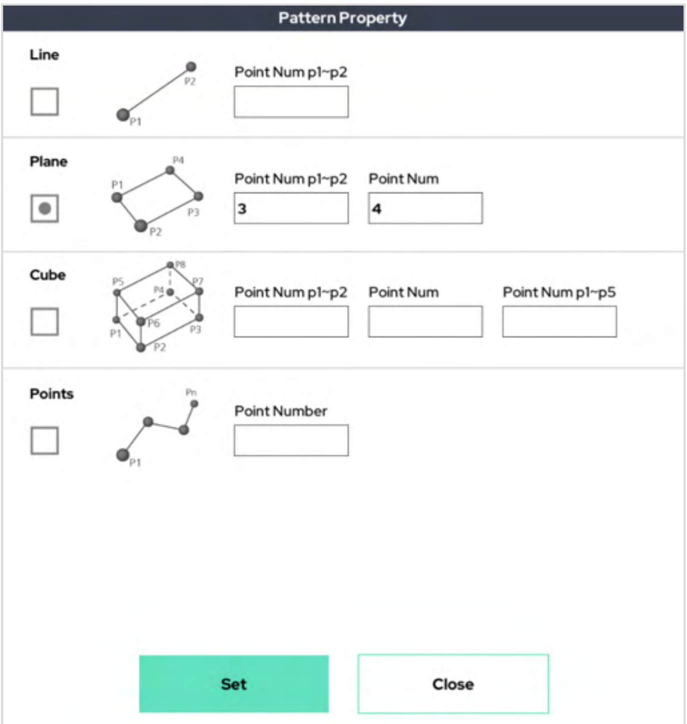
- Pattern Action:

  Pattern Anchor 에서 설정된 기준점에 대비하여 상대적인 동작을 정의하는 구역입니다. 정의된 상대적인 동작은, Pattern Property 에서 설정한 모든 패턴 지점에서 마다 반복됩니다.

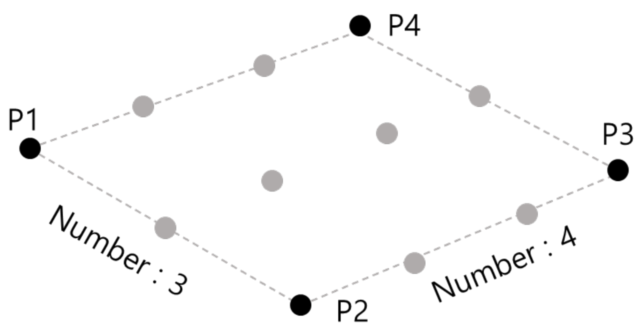
아래는 Pattern 기능의 예시입니다.



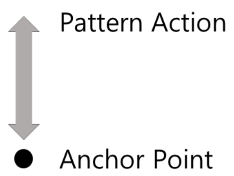
Step 1) Pattern Property 를 아래와 같이 세팅합니다.



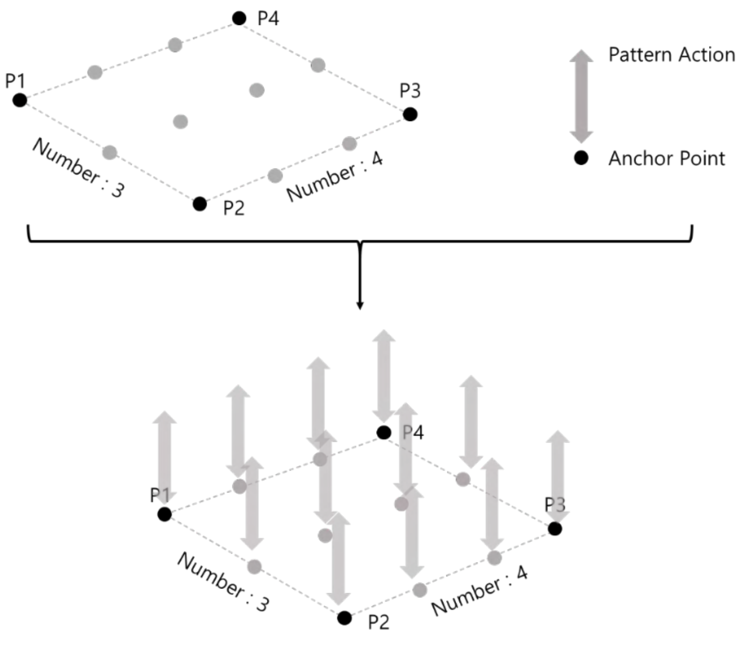
이 설정을 통해 아래와 반복 지점들이 공간상에 형성됩니다.



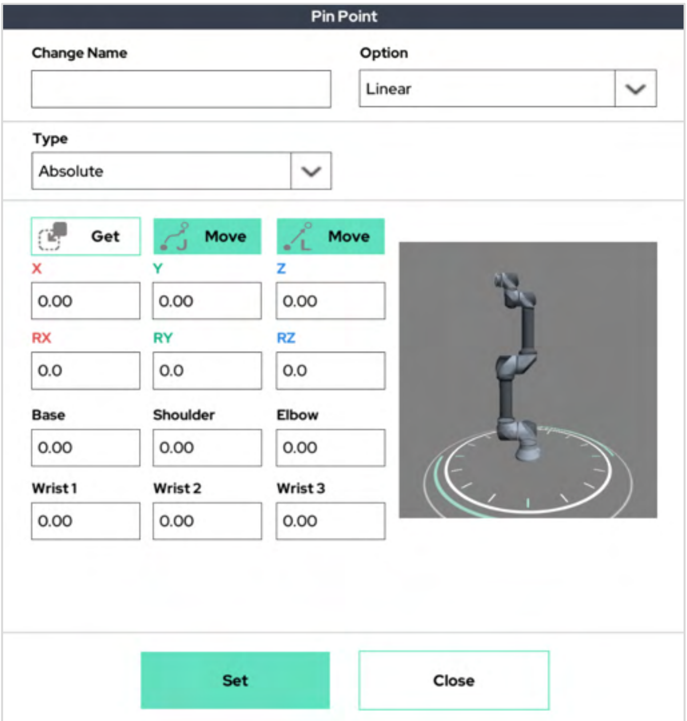
Step 2) Pattern Anchor 와 Pattern Action 을 통해 아래와 같은 상대 움직임을 정의합니다.



Step 3) 최종적으로 Step-2 에서 설정한 상대움직임이 Step-1 에서 설정된 패턴 지점들 모두에게 반영되면서 같은 동작을 반복하는 동작이 완성됩니다.



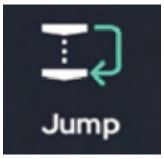
■ Pinpoint 기능 :

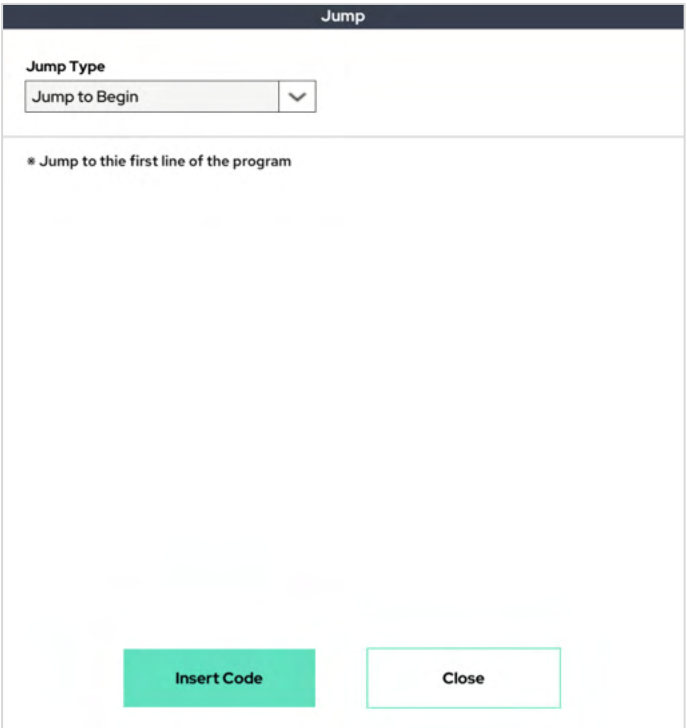


자세 정보 저장 만을 위한 특수 기능입니다. 특정 자세/위치의 정보를 Point 변수로 저장하기 위한 기능입니다. 특정 자세를 티칭해둔 상태에서 PinPoint 를 생성하고, PinPoint 의 이름을 부여하면, 해당 자세 정보는 Point 변수 형태로 변수화 됩니다.

Point 변수로 저장된 정보는, 다른 동작 명령어/설정 등에서 사용 가능합니다.

■ Jump 기능 :





프로그램 흐름을 불연속적으로 제어할 수 있는 기능입니다. 몇 가지 하위 옵션을 통하여 프로그램 흐름을 바꿀 수 있습니다.

옵션) Jump to Begin

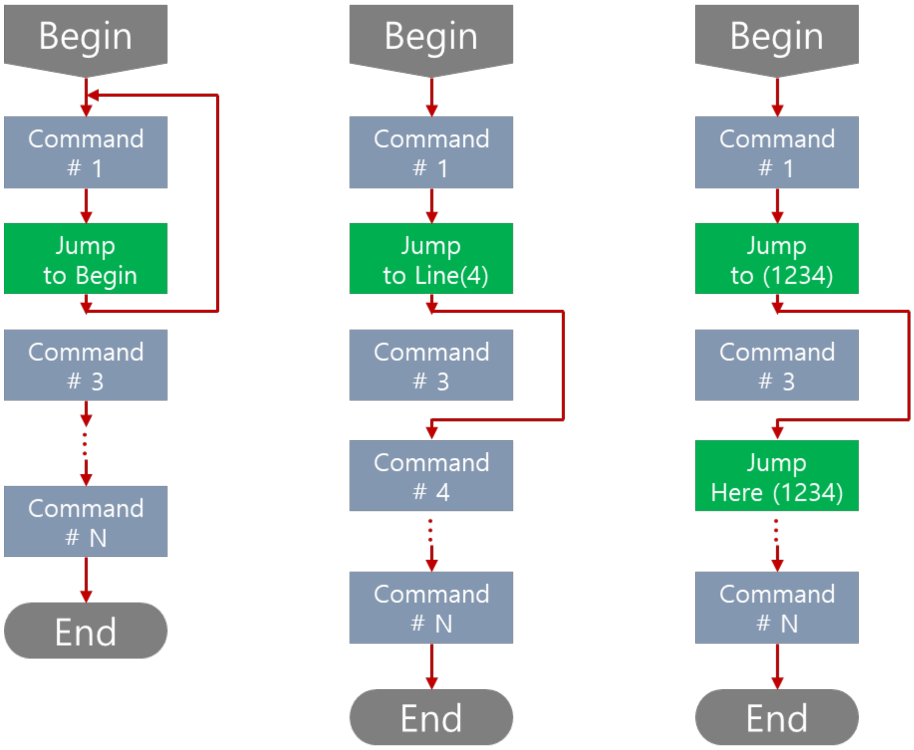
프로그램 흐름을 첫 줄로 이동시킵니다.

옵션) Jump to Line

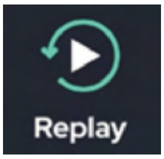
프로그램 흐름을 특정 줄 번호로 이동시킵니다.

옵션) Jump To / Here

프로그램 흐름을 특정 주소 값까지 이동시킵니다. JumpTo 에서는 이동하고자 하는 주소값을 호출하고, JumpHere 에서는 주소 값을 기입합니다.


Begin
Command # 1
Jump to Begin
Command # 3
Command # N
End
Begin
Command # 1
Jump to Line(4)
Command # 3
Command # 4
Command # N
End
Begin
Command # 1
Jump to (1234)
Command # 3
Jump Here (1234)
Command # N
End

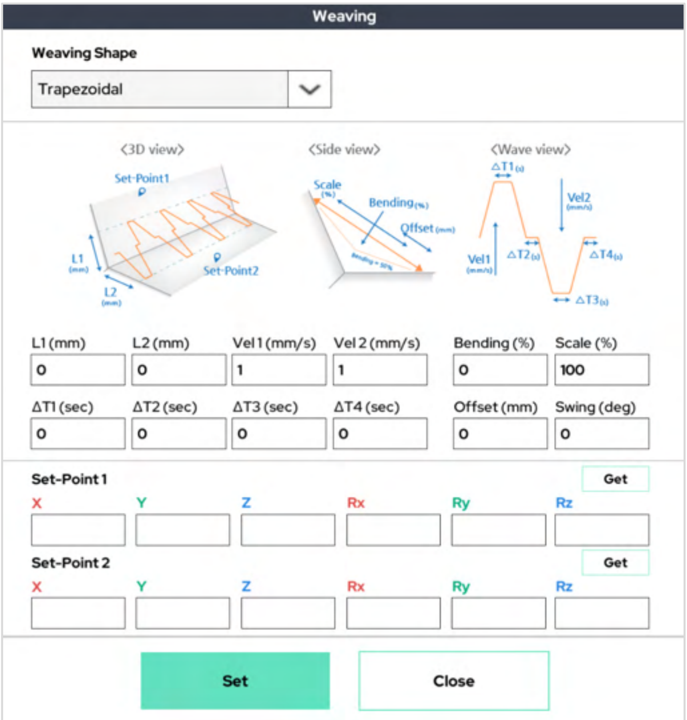
■ Replay 기능 :



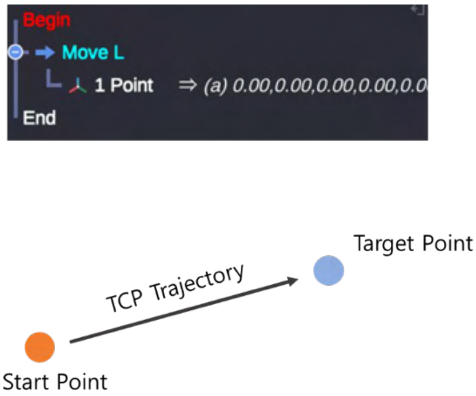
녹화된 티칭 모션을 재생하기 위한 기능입니다. 동작 녹화는 Make 페이지의 설정에서 수행합니다.
녹화된 모션의 이름과 동작 속도/속성을 선택하면, 녹화된 모션을 다시 재생합니다.

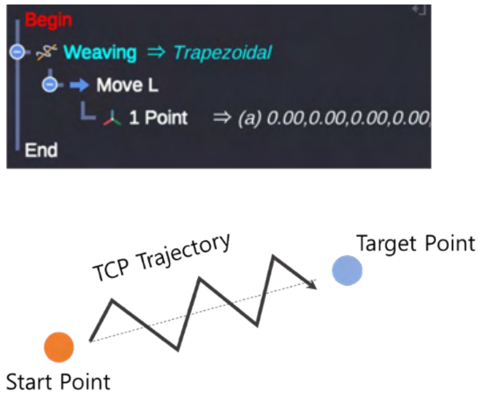
■ Weaving 기능 :

용접 위빙을 위한 특수 기능입니다. 위빙 기능 하위에 포함된 동작들은 설정된 위빙을 하도록 TCP 궤적이 자동으로 변경됩니다. 원하는 위빙의 형상과, 위빙 옵션을 선택 및 입력하면 됩니다.



아래 그림의 좌측은 일반적인 동작만 할 경우입니다. 이 동작을 그대로 위빙의 하위 항목으로 넣으면, 위빙 궤적이 반영된 TCP 궤적이 그려집니다(우측 그림 예시는 삼각파 위빙의 경우).


Begin
Move L
1 Point ⇒ (a) 0.00,0.00,0.00,0.00,0.0
End
TCP Trajectory
Target Point
Start Point

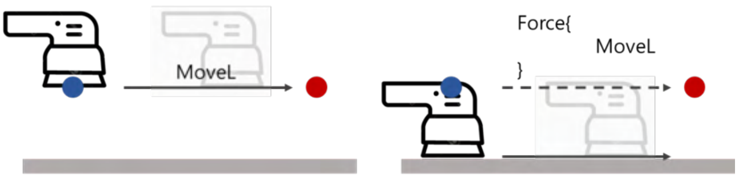

Begin
Weaving ⇒ Trapezoidal
Move L
1 Point ⇒ (a) 0.00,0.00,0.00,0.00
End
TCP Trajectory
Target Point
Start Point

■ Force 기능 :

힘 제어를 위한 기능입니다. 힘 제어 기능 하위에 포함된 동작들은 설정된 힘을 주도록 궤적이 자동으로 변경됩니다.

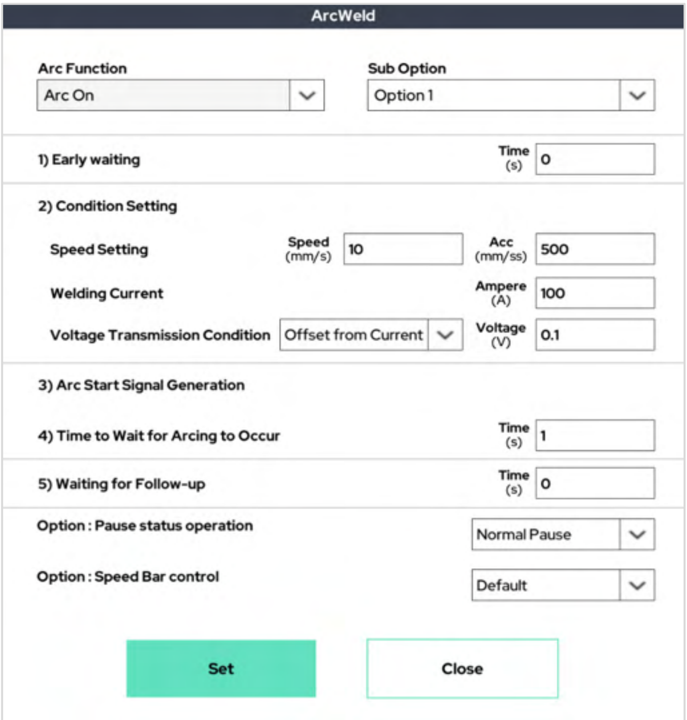
원하는 힘 제어 모드, 힘 제어에 사용할 센서, 그리고 힘 제어 대상 값을 선택 및 입력하면 됩니다.

아래 그림의 좌측은 일반적인 동작만 할 경우입니다. 평면 위 공중에서 동작이 시작되어 공중에서 끝나는 동작입니다. 이 동작을 그대로 힘 제어의 하위 항목으로 넣으면, 일정 힘으로 지면을 누르는 동작으로 바뀝니다(힘 제어 세팅을 지면 방향으로 할 경우).



[힘 제어 적용 전]　　[힘 제어를 함께 사용할 경우]

■ Arcweld 기능 :

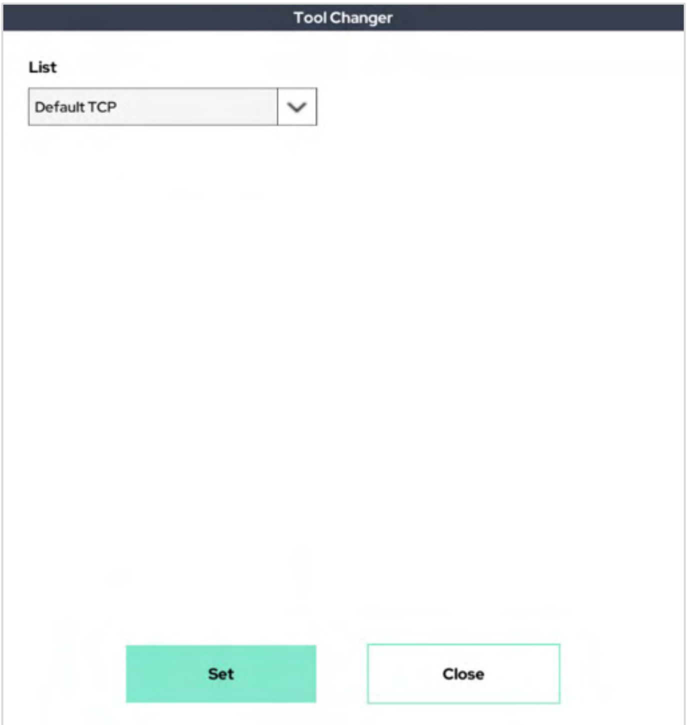


아크 용접을 위한 특수 기능입니다. Wait / D.out 등을 이용하여 구현 가능한 기능을 빠르게 사용 가능하도록 만들어진 특수 매크로 기능입니다.

이 기능을 사용하기 위해서는 Setup 페이지의 Device 란에서 용접기에 관한 파라미터 및 연결 정보 설정이 선행되어야 합니다.

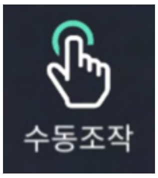
위의 그림처럼, 이 기능을 사용하면, 용접 시 사용할 용접 속도/ 용접 전류 / 전압 설정 / 안전 신호 처리 옵션 등을 빠르고 쉽게 프로그램에 삽입 가능합니다.

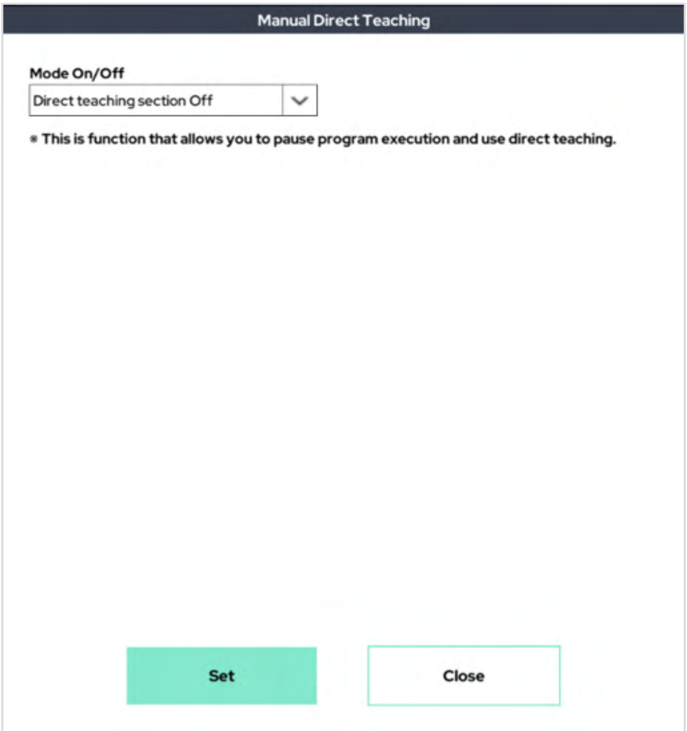
■ TCP 설정 기능 :



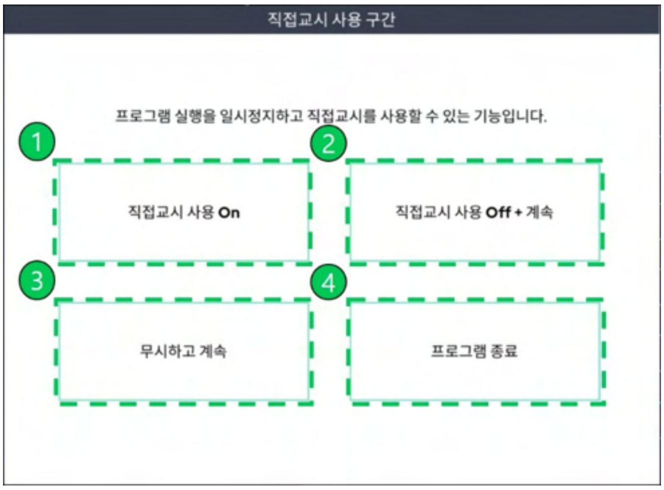
Setup 의 Tool List 에 미리 저장해 놓은 TCP 값으로 프로그램 실행 중 TCP 값을 변경하는 기능입니다. 다시 TCP 값을 바꾸거나 프로그램이 종료되기 전까지 변경되지 않습니다.

■ 수동조작 기능 :





프로그램 실행 도중 직접 교시 기능을 사용할 가능하게 하는 기능입니다. 모드 On 할 경우 수동 조작 명령 실행 시 프로그램은 일시 정지되며 아래 그림과 같은 팝업 창이 화면에 나오게 됩니다.



팝업 창에서 4 개의 기능을 선택할 수 있습니다.

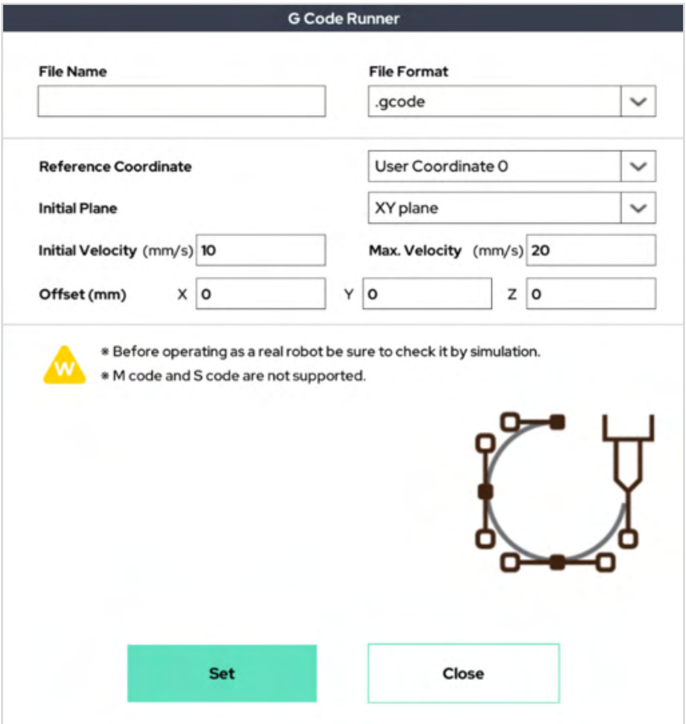
① 프로그램이 일시 정지된 상태에서 직접 교시 기능을 사용합니다.

② ①에서 직접 교시 기능을 사용했을 경우 직접 교시 기능을 끄고 프로그램을 재개합니다.

③ 수동 조작 기능을 무시하고 프로그램을 재개합니다.

④ 프로그램을 종료합니다.

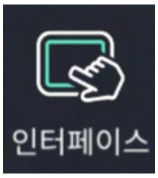
■ G 코드 기능 :



로봇이 G code 에 저장된 경로대로 움직이도록 하는 기능입니다. 지정된 경로의 폴더(\Tablet\Android\data\com.rainbow.cobot\files\work)에 G code 파일을 미리 저장해 놓아야 사용 가능한 기능입니다.

File Name 에 저장한 G code 파일의 이름을 입력합니다. 이때 로봇이 움직이는 평면은 사용자 지정 좌표계의 xy, yz, zx 평면을 시작 평면으로 지정할 수 있다.

■ 인터페이스 기능 :

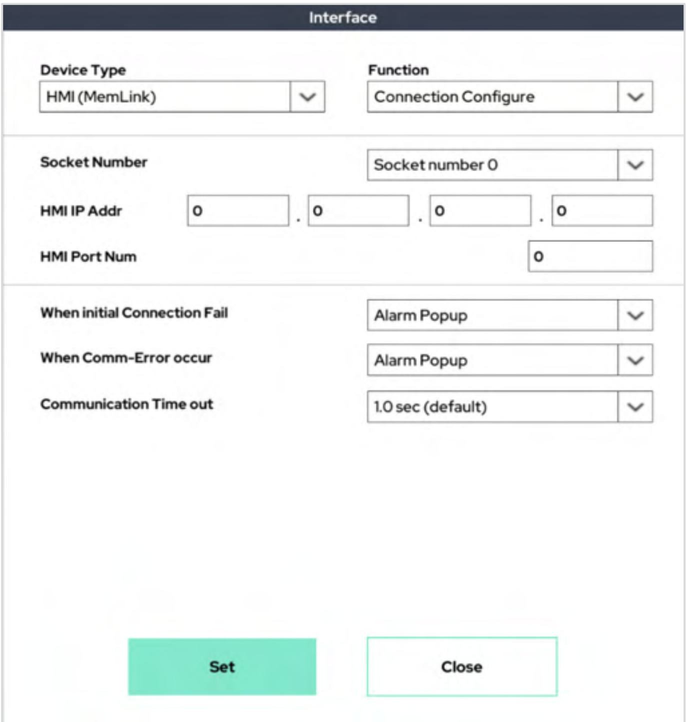


인터페이스 기능은 PLC, HMI, PC 등과 같은 외부 기기를 제어 박스와 연결하기 위한 기능입니다. 인터페이스를 이용하여 사용할 수 있는 외부 기기의 리스트는 다음과 같습니다.

- HMI(MemLink) - Proface, TOP
- PLC(MC Protocol) – 미쯔비시 PLC
- Mulic Player
- PLC(XGT Protocol) - LS Electric PLC
- CSV File
- Pickit
- Modbus Client(Interrupt)

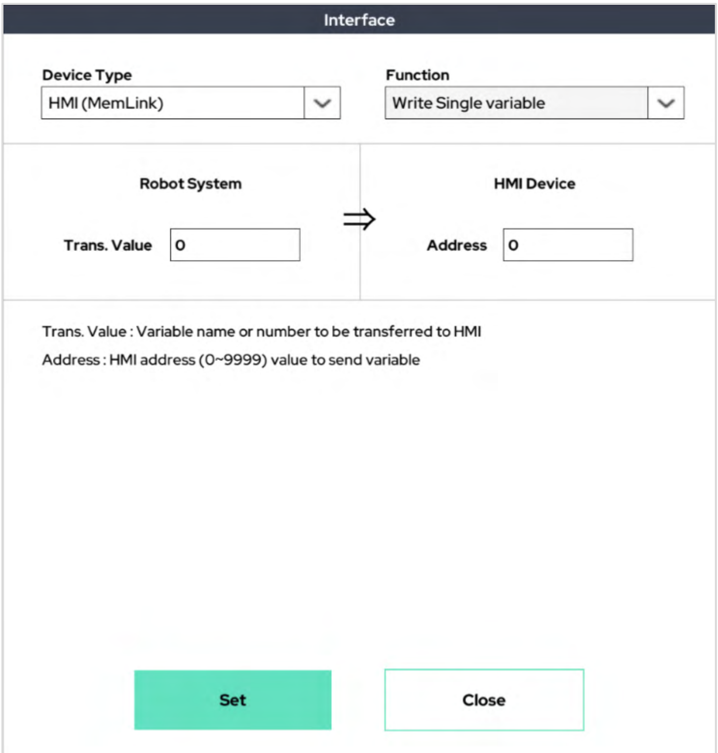
각각의 외부 기기에 따라 사용할 수 있는 세부 기능이 다르기 때문에 하기 내용 참고하여 사용해야 합니다.

HMI(MemLink)-연결설정



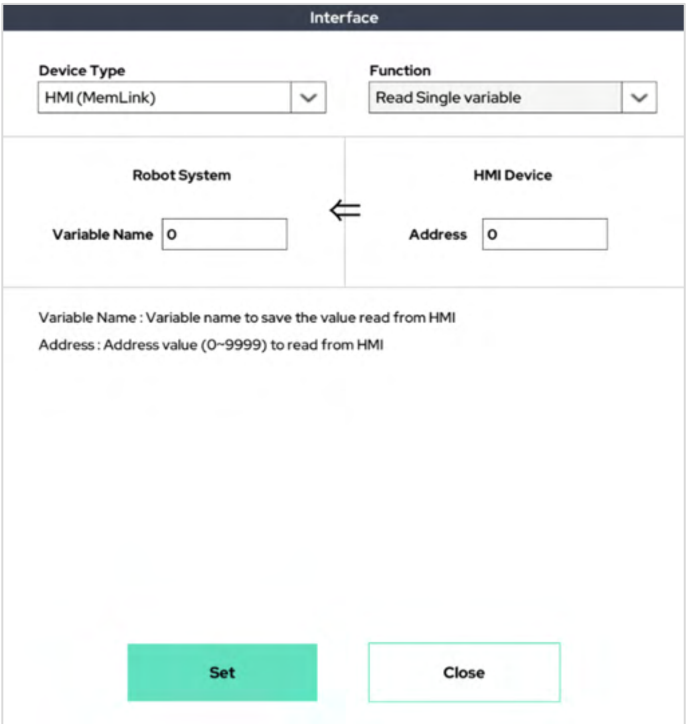
HMI 와 RB 시스템 사이에 통신을 연결하는 창입니다. 소켓 번호, IP 주소 및 포트를 입력해주게 됩니다. 또한, 연결 실패, 통신 에러 발생시 알람 팝업을 띄울지 무시할지를 결정할 수 있고 통신 타임 아웃 시간을 정할 수 있습니다.

HMI(MemLink)-단일 변수 쓰기



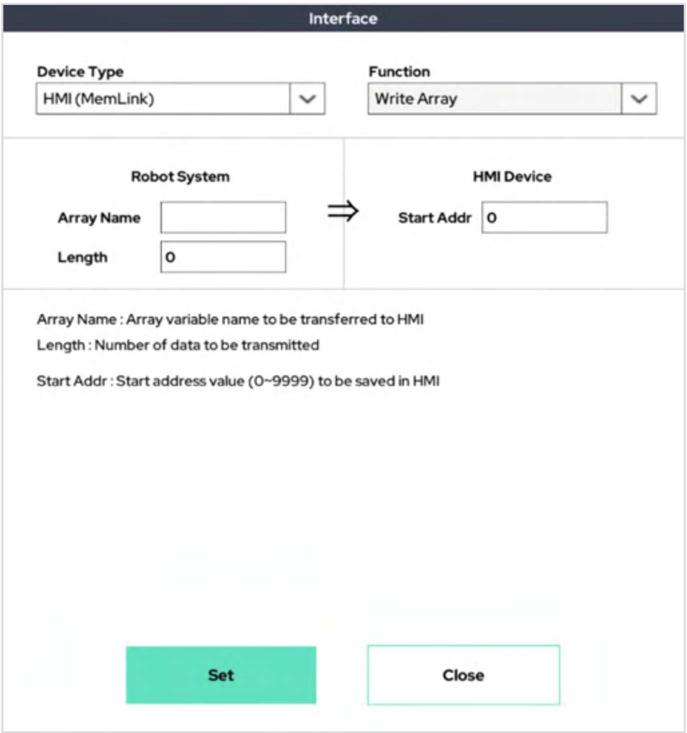
HMI 의 1 개의 주소에 값을 입력하는 기능입니다. 이때 전송 값은 숫자 또는 변수(Variable)의 이름을 입력합니다.

HMI(MemLink)-단일 변수 읽기



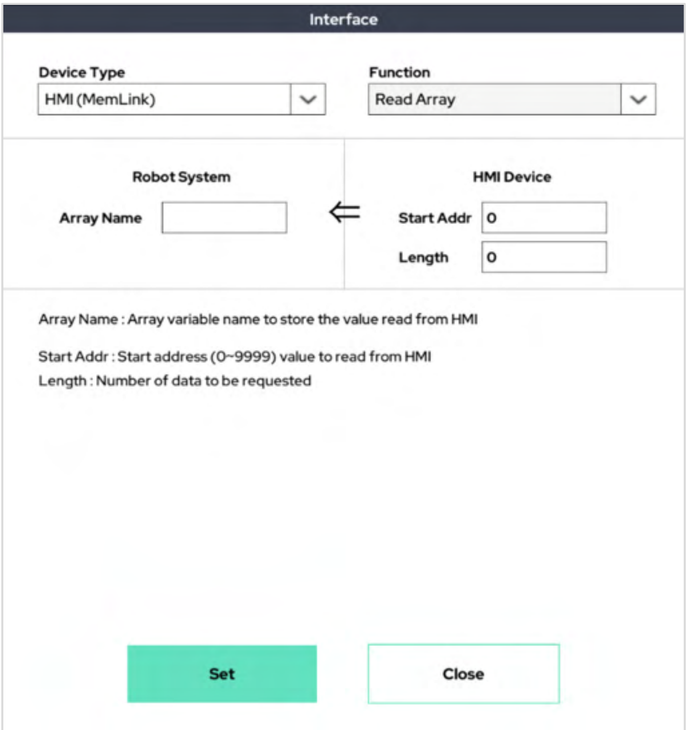
HMI 의 1 개의 주소에서 값을 읽어오는 기능입니다. 이때 읽은 값은 지정한 변수(Variable)에 저장하게 됩니다.

HMI(MemLink)-다중 숫자(배열) 쓰기



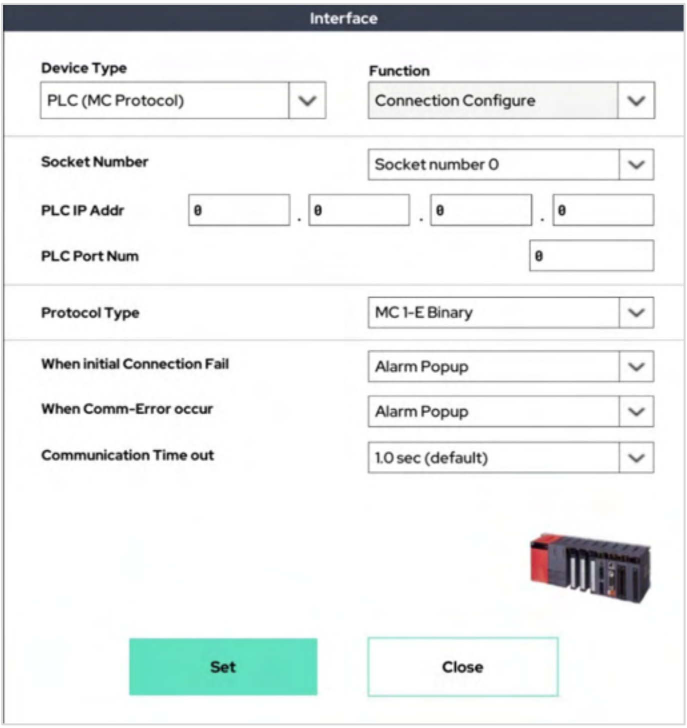
HMI 의 시작 주소부터 지정한 개수의 주소까지 숫자를 입력하는 기능입니다. 이때 미리 선언한 배열을 Array Name 에 적어줘야 하고 길이는 배열의 최대 길이인 20 을 넘지 않아야 합니다.

HMI(MemLink)-다중 숫자(배열) 읽기



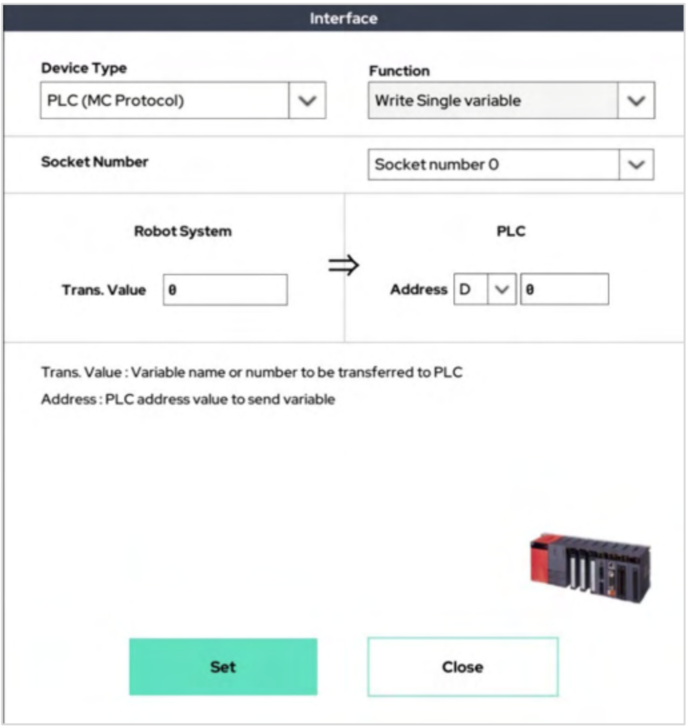
HMI 의 시작 주소부터 지정한 개수의 주소까지 데이터를 읽어오는 기능입니다. 이때 미리 선언한 배열을 Array Name 에 적어줘야 하고 길이는 배열의 최대 길이인 20 을 넘지 않아야 합니다.

PLC(MC Protocol)-연결 설정



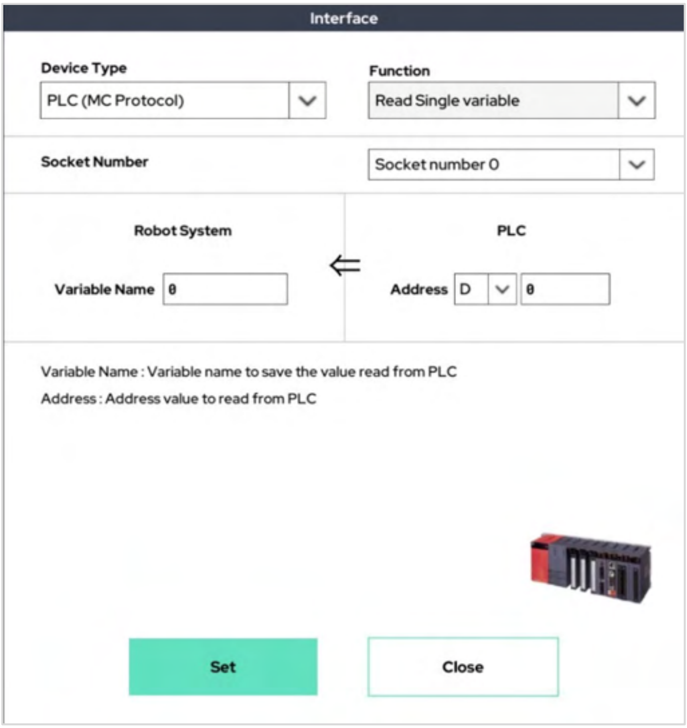
미쯔비시 PLC 와 RB 시스템 사이에 통신을 연결하는 창입니다. 소켓 번호, IP 주소 및 포트를 입력해주게 됩니다. 또한, 프로토콜 타입, 연결 실패, 통신 에러 발생시 알람 팝업을 띄울지 무시할지를 결정할 수 있고 통신 타임 아웃 시간을 정할 수 있습니다.

PLC(MC Protocol)-단일 변수 쓰기



PLC 의 1 개의 주소에 값을 입력하는 기능입니다. 이때 전송 값은 숫자 또는 변수(Variable)의 이름을 입력합니다.

PLC(MC Protocol)- 단일 변수 읽기

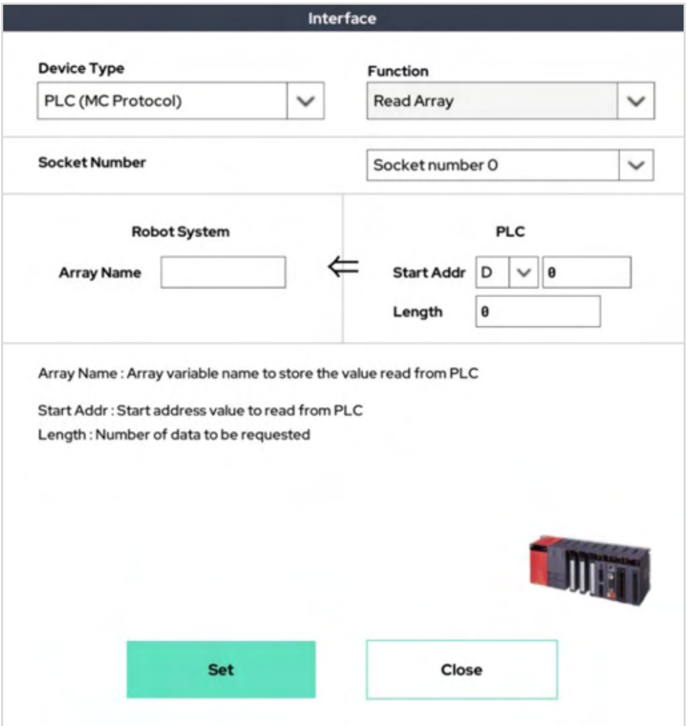


PLC 의 1 개의 주소에서 값을 읽어오는 기능입니다. 이때 읽은 값은 지정한 변수(Variable)에 저장하게 됩니다.

PLC(MC Protocol)- 다중 숫자(배열) 쓰기

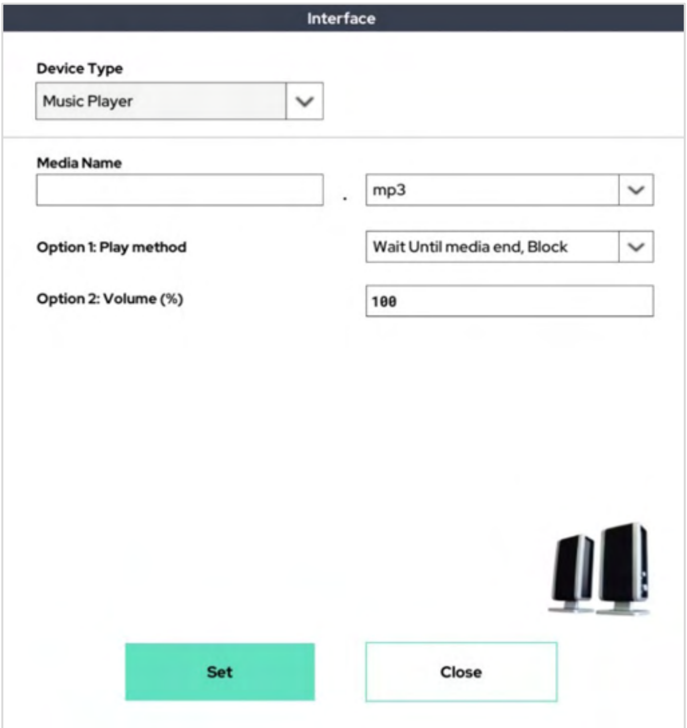
PLC 의 시작 주소부터 지정한 개수의 주소까지 숫자를 입력하는 기능입니다. 이때 미리 선언한 배열을 Array Name 에 적어줘야 하고 길이는 배열의 최대 길이인 20 을 넘지 않아야 합니다.

PLC(MC Protocol)- 다중 숫자(배열) 읽기



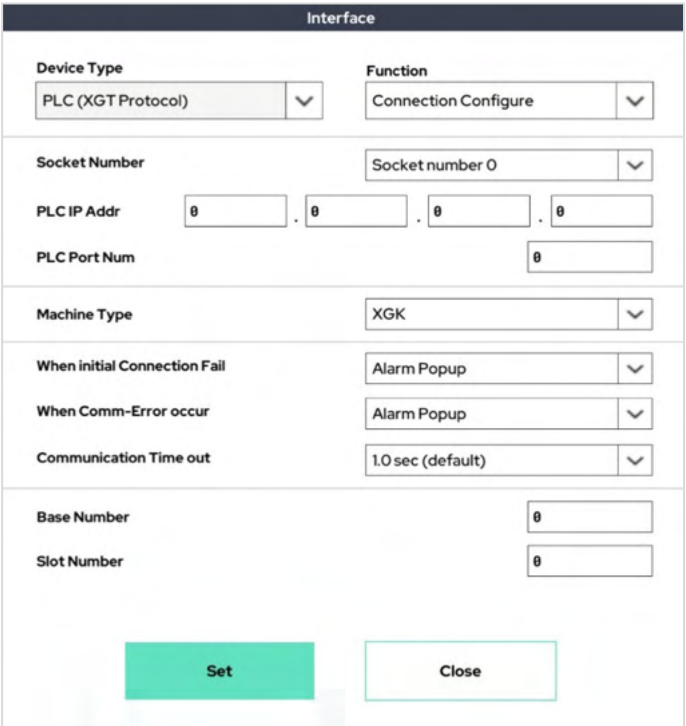
PLC 의 시작 주소부터 지정한 개수의 주소까지 데이터를 읽어오는 기능입니다. 이때 미리 선언한 배열을 Array Name 에 적어줘야 하고 길이는 배열의 최대 길이인 20 을 넘지 않아야 합니다.

Music Player



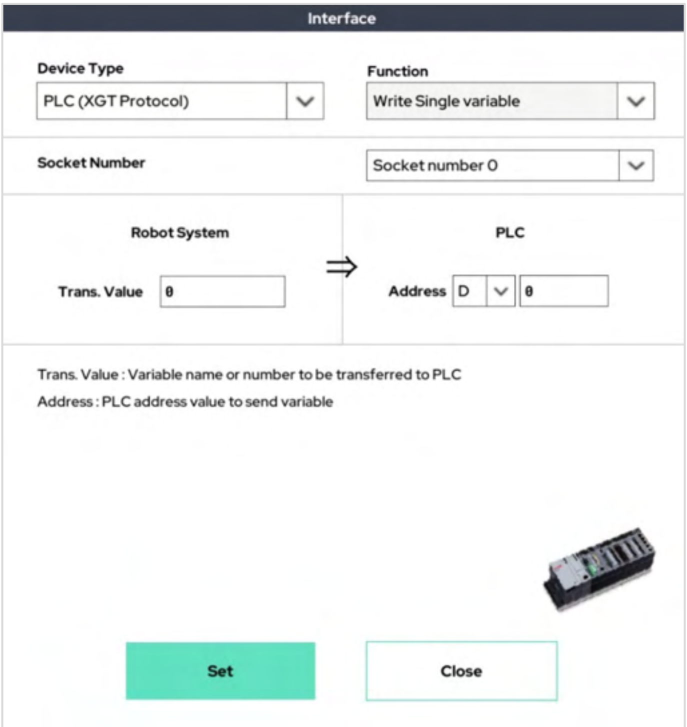
프로그램 재생 중에 mp3 파일을 재생하는 기능입니다. RB Driver 를 통해 Music 드라이버를 설치해야 하고, 재생하고자 하는 mp3 파일이 정해진 경로 내에 존재해야 합니다.

PLC(XGT Protocol)-연결 설정



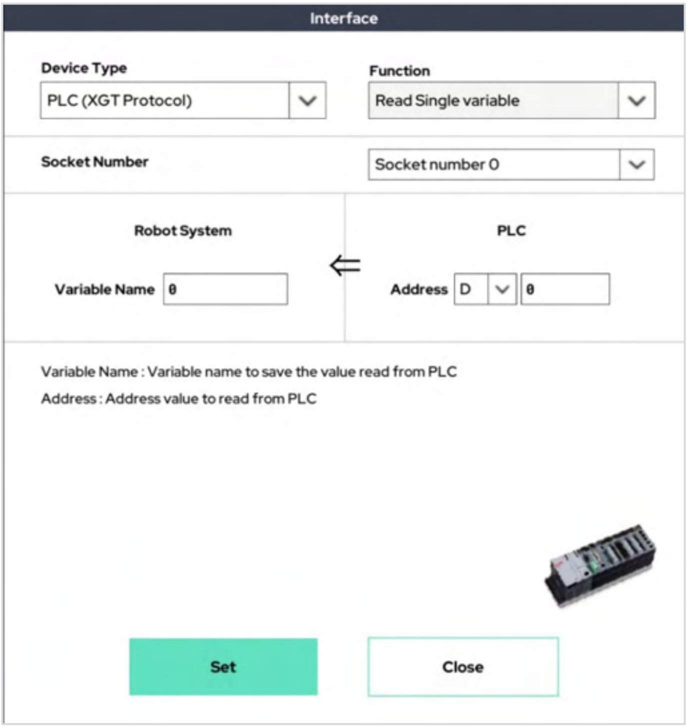
LS Electric PLC 와 RB 시스템 사이에 통신을 연결하는 기능입니다. 소켓 번호, IP 주소 및 포트를 입력해주게 됩니다. 또한, 프로토콜 타입, 연결 실패, 통신 에러 발생시 알람 팝업을 띄울지 무시할지를 결정할 수 있고 통신 타임 아웃 시간을 정할 수 있습니다.

PLC(XGT Protocol)-단일 변수 쓰기



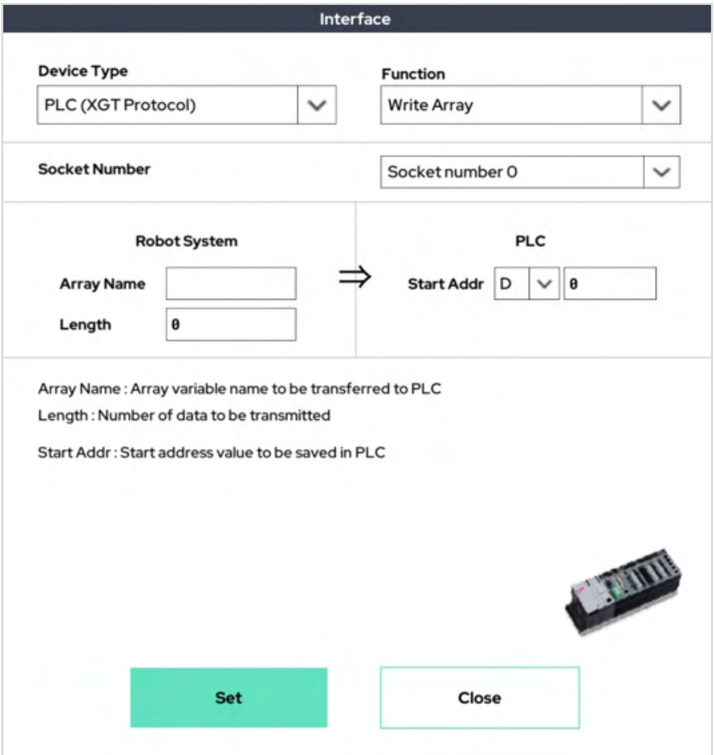
PLC 의 1 개의 주소에 값을 입력하는 기능입니다. 이때 전송 값은 숫자 또는 변수(Variable)의 이름을 입력합니다.

PLC(XGT Protocol)-단일 변수 읽기



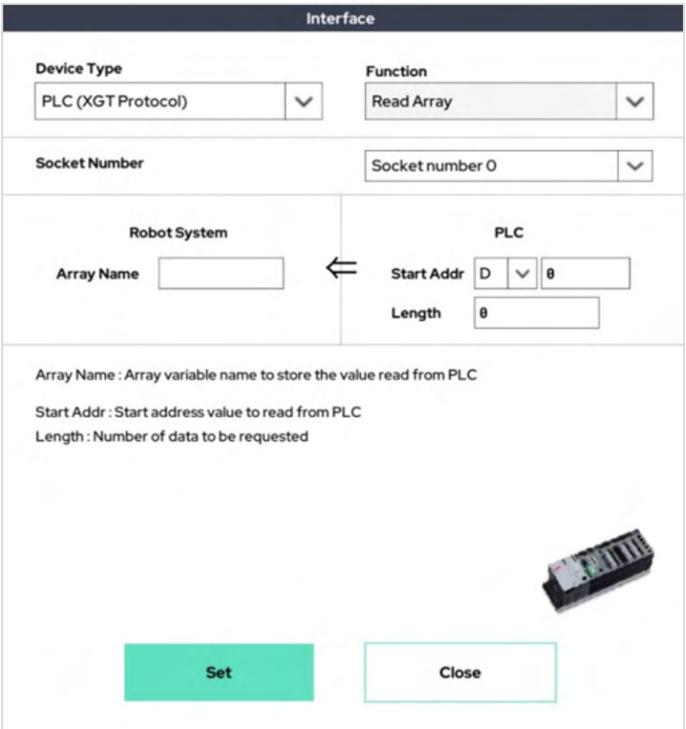
PLC 의 1 개의 주소에서 값을 읽어오는 기능입니다. 이때 읽은 값은 지정한 변수(Variable)에 저장하게 됩니다.

PLC(XGT Protocol)-다중 숫자(배열) 쓰기



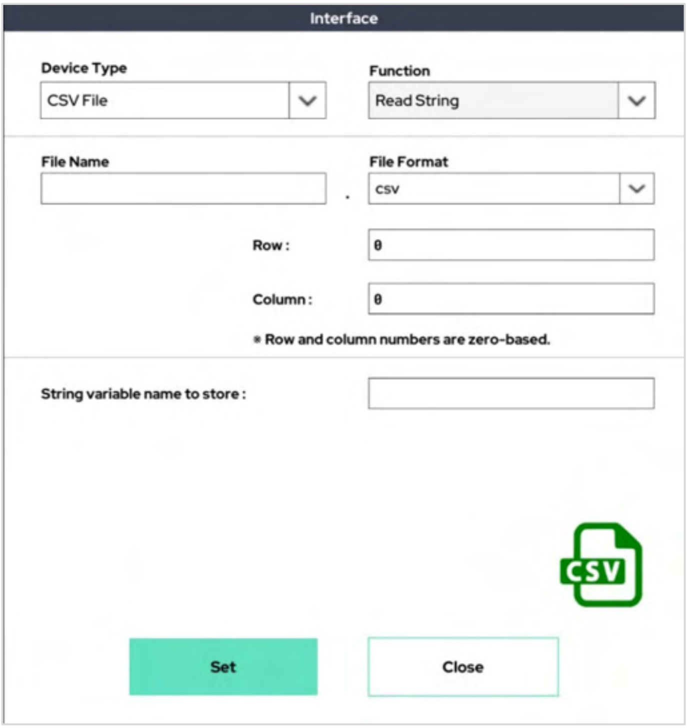
PLC 의 시작 주소부터 지정한 개수의 주소까지 숫자를 입력하는 기능입니다. 이때 미리 선언한 배열을 Array Name 에 적어줘야 하고 길이는 배열의 최대 길이인 20 을 넘지 않아야 합니다.

PLC(XGT Protocol)-다중 숫자(배열) 읽기



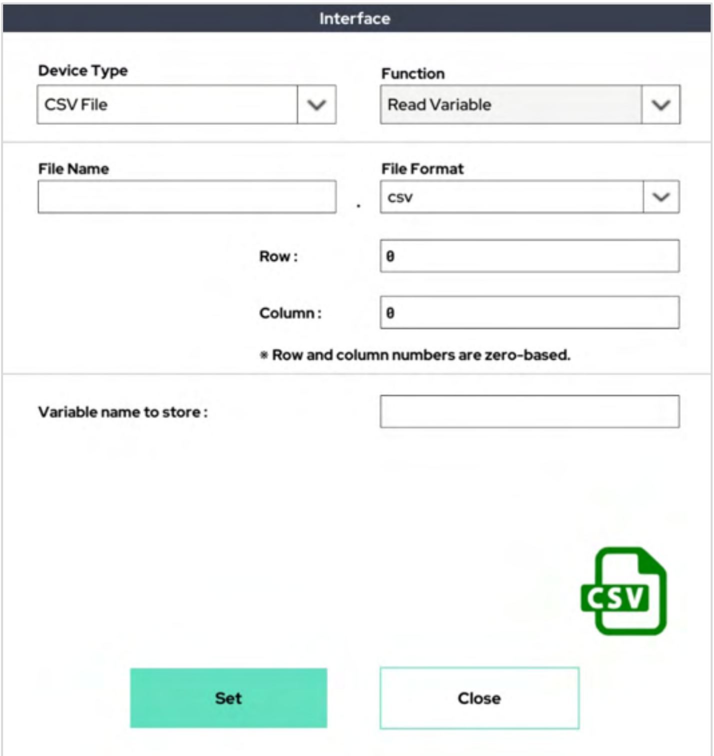
PLC 의 시작 주소부터 지정한 개수의 주소까지 데이터를 읽어오는 기능입니다. 이때 미리 선언한 배열을 Array Name 에 적어줘야 하고 길이는 배열의 최대 길이인 20 을 넘지 않아야 합니다.

CSV File-문자열 읽기



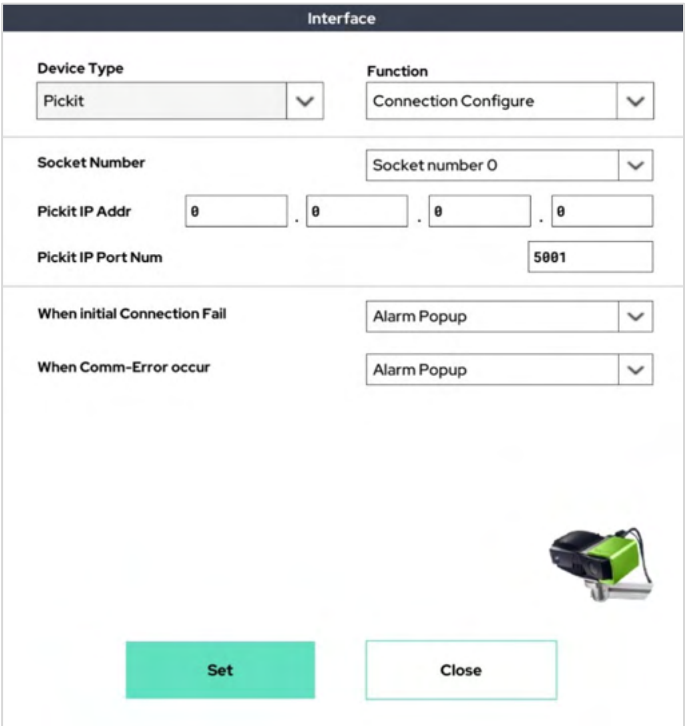
CSV File 로부터 문자열을 읽어오는 기능입니다. 해당 CSV File 은 지정한 경로 내에 저장되어 있어야 합니다.

CSV File-단일 숫자 읽기



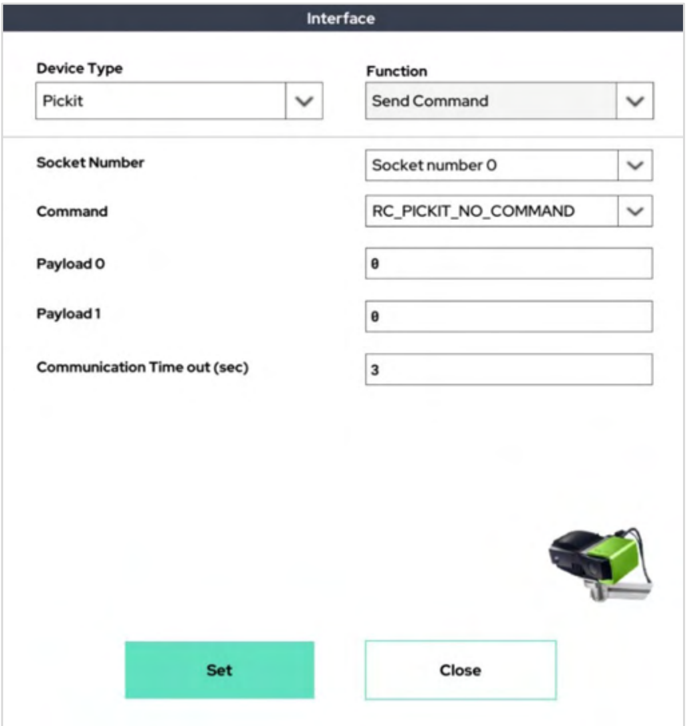
CSV File 로부터 단일 숫자를 읽어오는 기능입니다. 해당 CSV File 은 지정한 경로 내에 저장되어 있어야 합니다.

Pickit-연결 설정



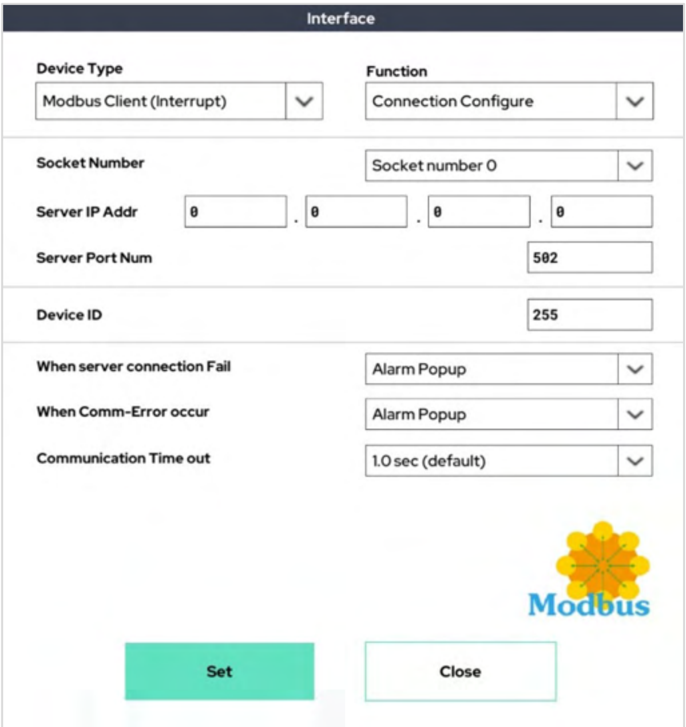
Pickit 과 RB 시스템 사이에 통신을 연결하는 기능입니다. 소켓 번호, IP 주소를 입력해주게 됩니다. 또한, 연결 실패, 통신 에러 발생시 알람 팝업을 띄울지 무시할지를 결정할 수 있습니다.

Pickit -명령어 전송



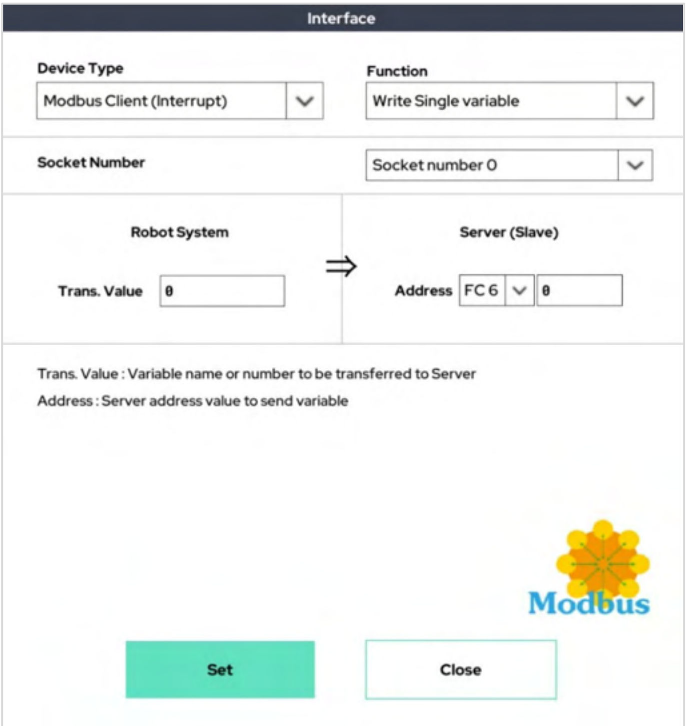
Pickit 으로 전송할 명령, 명령에 따른 데이터를 설정합니다.

Modbus Client(Interrupt) -연결 설정



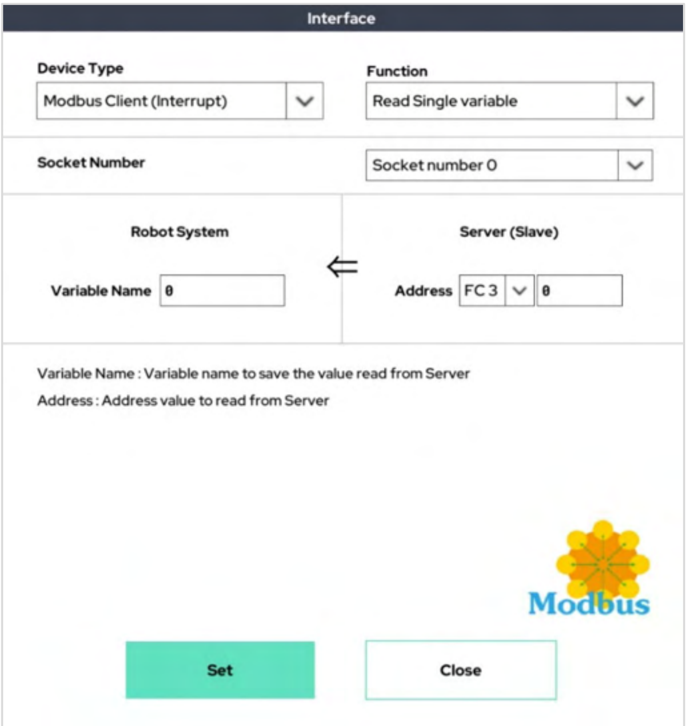
모드버스 통신에서 RB 시스템을 클라이언트로 연결하는 기능입니다. 소켓 번호, IP 주소를 입력해주게 됩니다. 또한, 연결 실패, 통신 에러 발생시 알람 팝업을 띄울지 무시할지를 결정할 수 있습니다.

Modbus Client(Interrupt) -단일 변수 쓰기



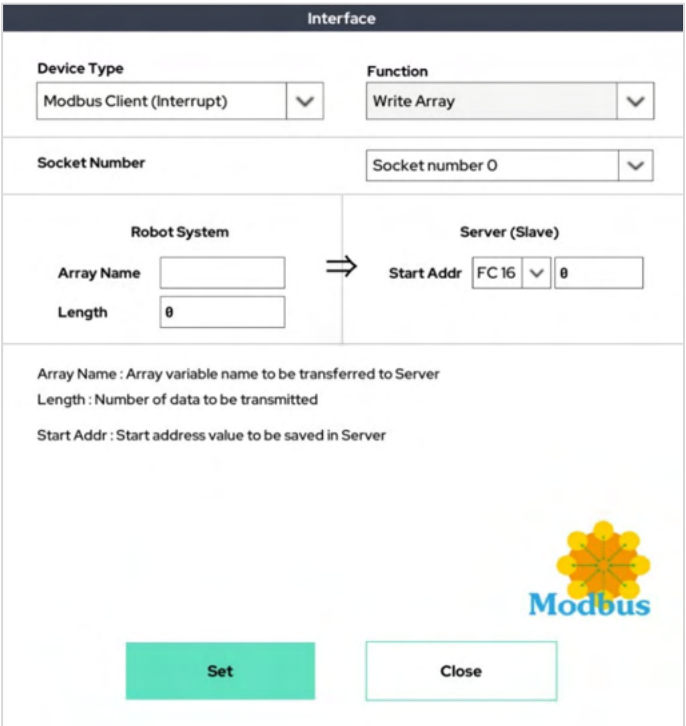
모드버스 통신을 통해 워드 타입 데이터를 1 개의 주소에 값을 입력하는 기능입니다. 이때 전송 값은 숫자 또는 변수(Variable)의 이름을 입력합니다.

Modbus Client(Interrupt) -단일 숫자 읽기



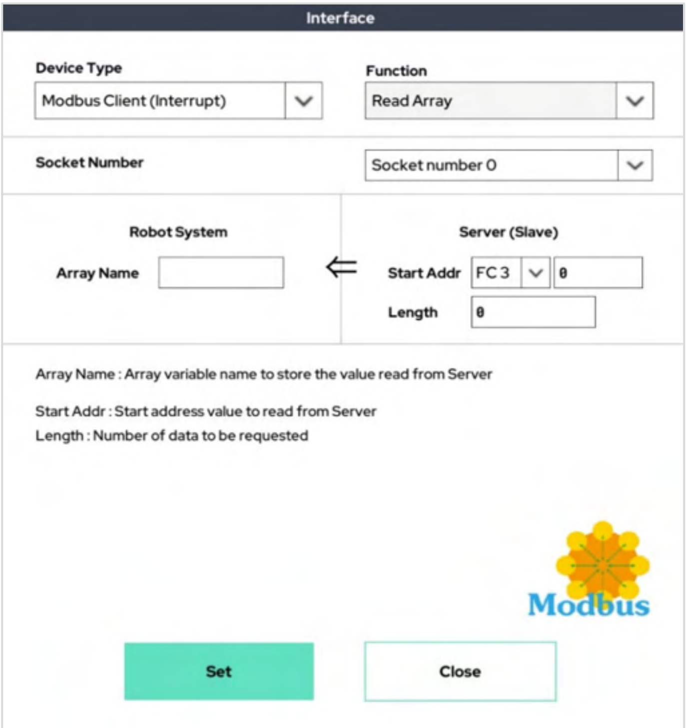
모드버스 통신을 통해 워드 타입 데이터 1 개의 값을 해당 주소에서 읽어오는 기능입니다. 이때 읽은 값은 지정한 변수(Variable)에 저장하게 됩니다.

Modbus Client(Interrupt) -다중 숫자(배열) 쓰기



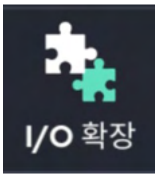
모드버스 통신을 통해 워드 타입 숫자 데이터를 시작 주소부터 지정한 개수의 주소까지 입력하는 기능입니다. 이때 미리 선언한 배열을 Array Name 에 적어줘야 하고 길이는 배열의 최대 길이인 20 을 넘지 않아야 합니다.

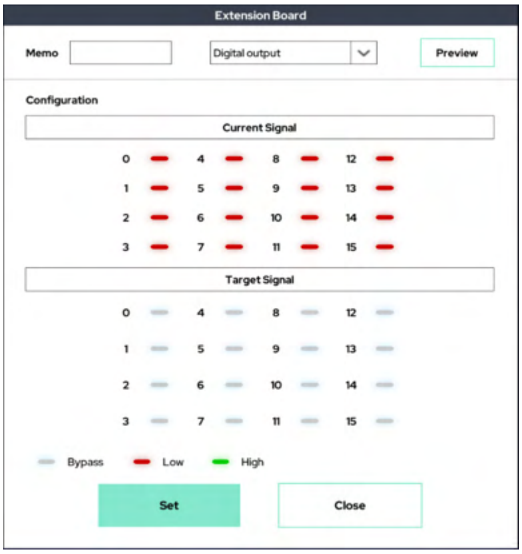
Modbus Client(Interrupt) -다중 숫자(배열) 읽기



모드버스 통신을 통해 시작 주소부터 지정한 개수의 주소까지 데이터를 읽어오는 기능입니다. 이때 미리 선언한 배열을 Array Name 에 적어줘야 하고 길이는 배열의 최대 길이인 20 을 넘지 않아야 합니다.

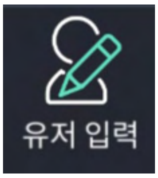
■ I/O 확장 기능 :

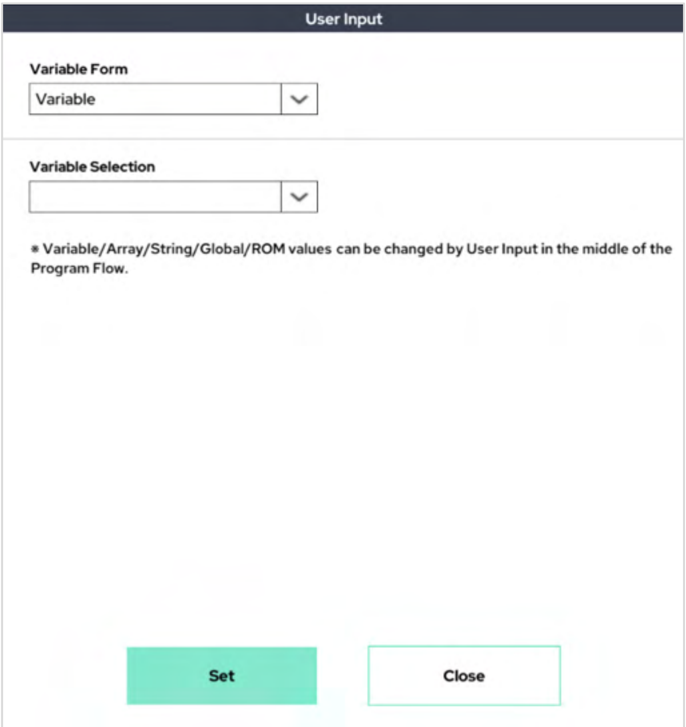




확장 I/O 모듈을 구매하여 사용할 경우 디지털/아날로그 출력을 제어하는 기능입니다. 기존의 D.출력, An.출력과 사용 방법은 동일합니다.

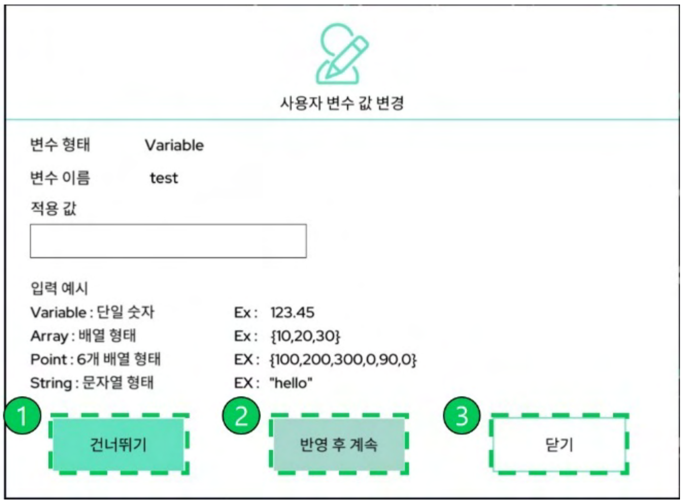
■ 유저 입력 기능 :





프로그램 실행 중 사용자가 특정 변수의 값을 임의로 변경하고 싶을 때 사용하는 기능입니다.
Variable/Array/Point/String/Global/ROM 변수에 대해 사용할 수 있습니다.

실행할 경우 아래와 같은 팝업 창이 뜨게 됩니다.



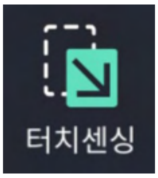
팝업 창에서 3 개의 기능을 선택할 수 있습니다.

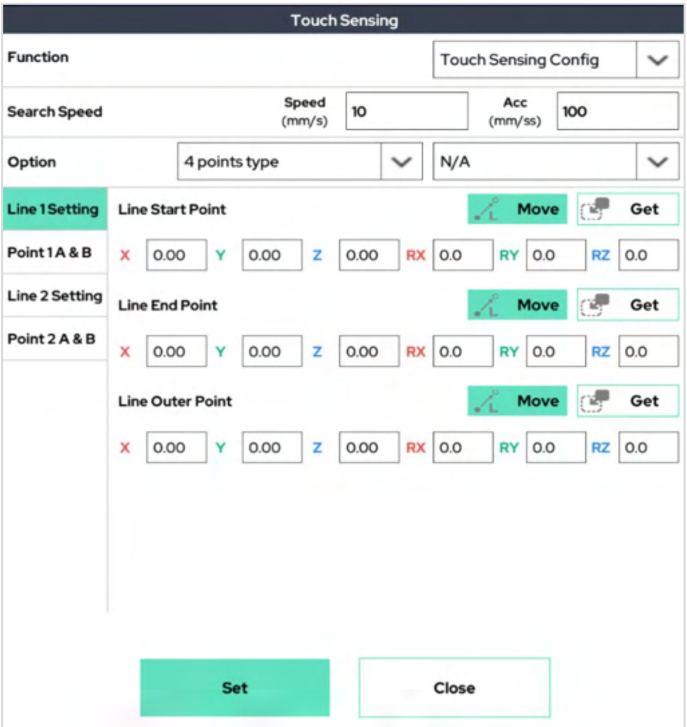
① 해당 변수를 바꾸지 않고 프로그램을 재개합니다.

② ‘적용 값’에 변경할 데이터를 입력한 후 누르게 되면 ‘적용 값’에 입력한 데이터가 반영된 후 프로그램이 재개됩니다.

③ 프로그램을 종료합니다.

■ 터치 센싱 기능 :



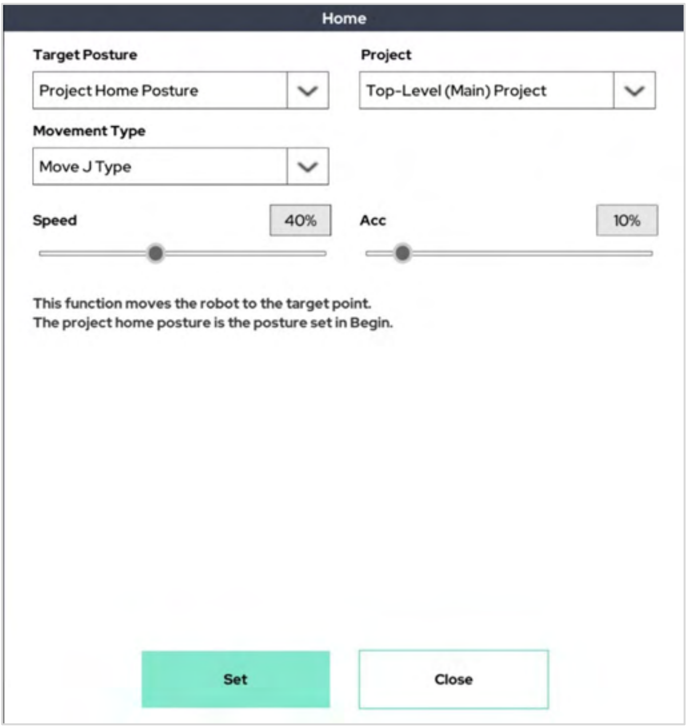


터치 센싱 기능은 용접 어플리케이션 활용을 위하 기능입니다.

모재의 이동을 감지해서 모재의 이동 방향을 반영하여 용접 시에 활용되게 됩니다.

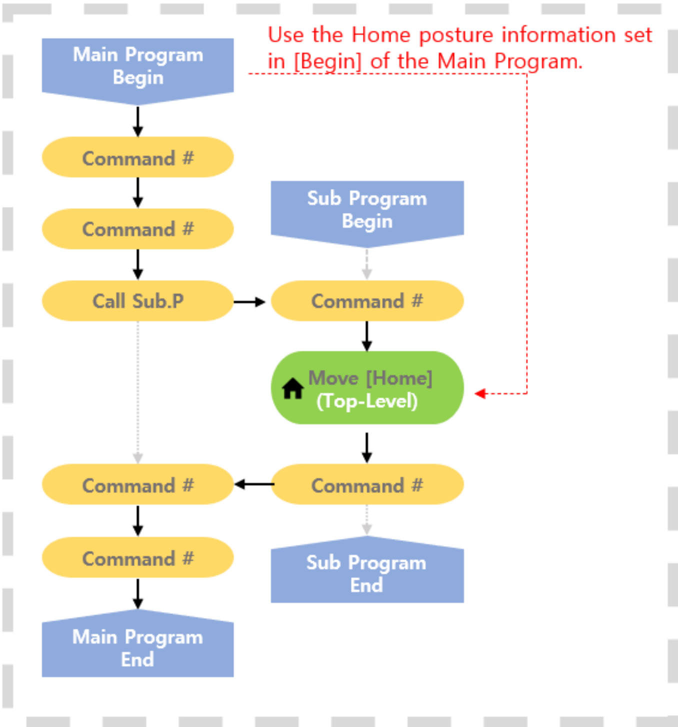
이 기능에 대한 자세한 설명은 별도의 매뉴얼로 제공됩니다.

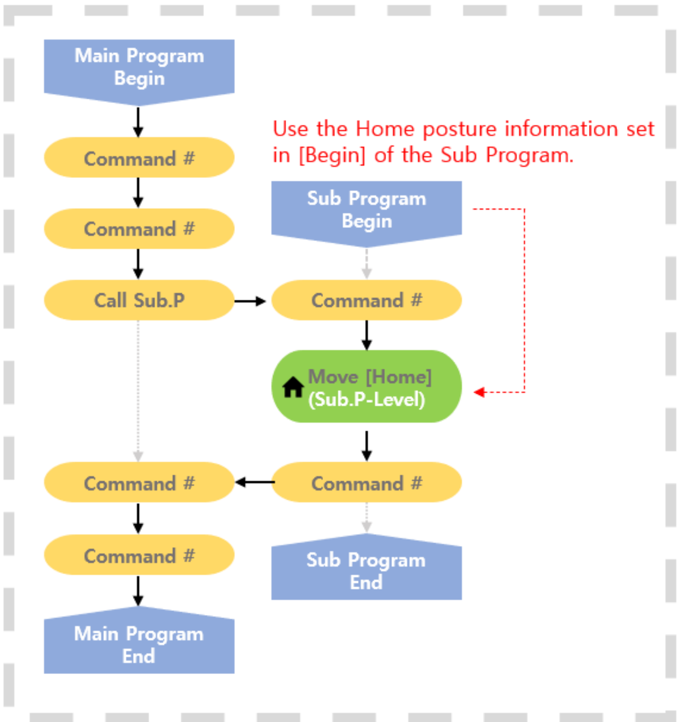
■ 홈 :



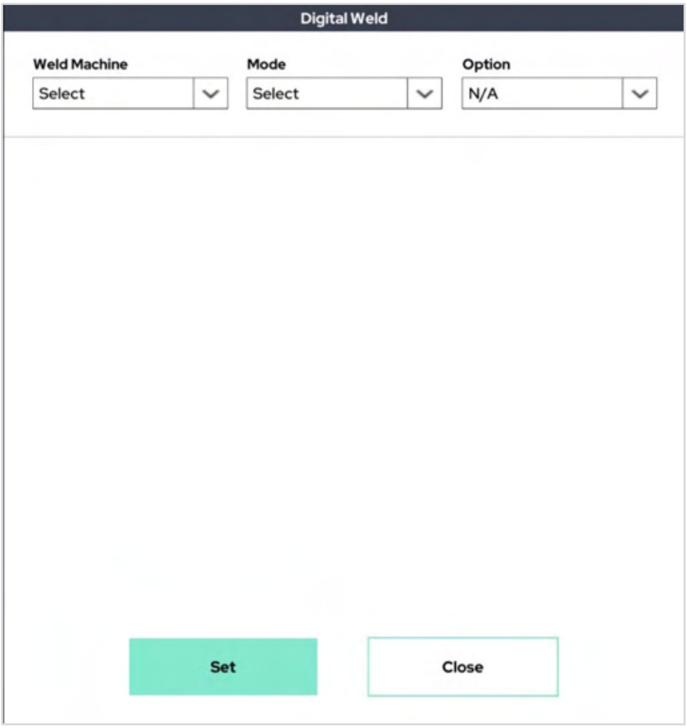
홈 기능은 프로젝트 시작 위치 또는 관절 0 점 자세로 로봇을 동작시키는 기능입니다. 이때, J 타입과 L 타입 동작을 선택할 수 있습니다. 프로젝트 시작 위치로 가는 경우 메인 프로그램의 시작 위치와 서브 프로그램의 시작 위치를 선택할 수 있습니다.

아래 다이어그램을 통해 서브 프로그램 내에서 홈 기능을 사용할 때 메인 프로젝트의 시작 위치로 가는 경우와 서브 프로젝트의 시작 위치로 가는 경우의 차이를 알 수 있습니다.



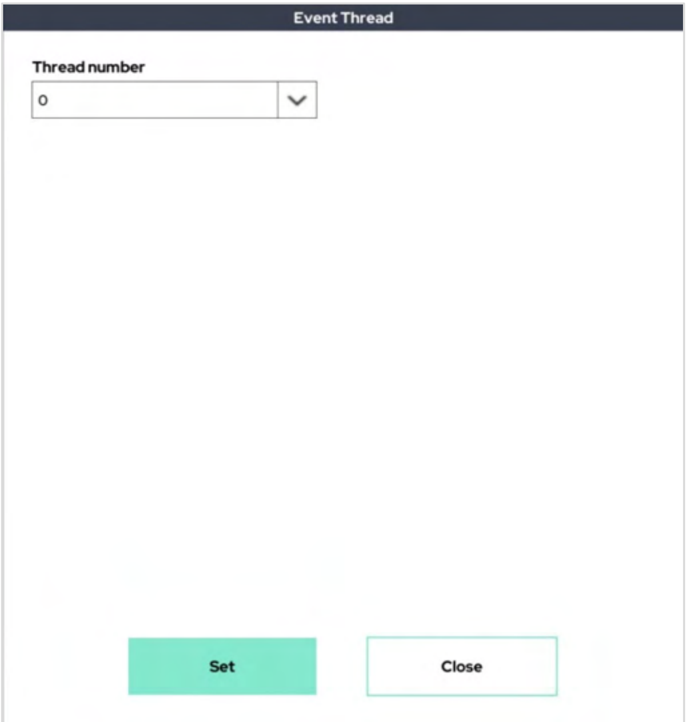


■ 용접기 :



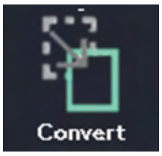
디지털 용접기를 사용할 수 있는 기능입니다. 사용하는 용접기를 선택한 후 용접 시작 조건, 용접 종료 조건, 용접 세팅 등을 진행할 수 있습니다.

■ 이벤트 쓰레드 호출 :



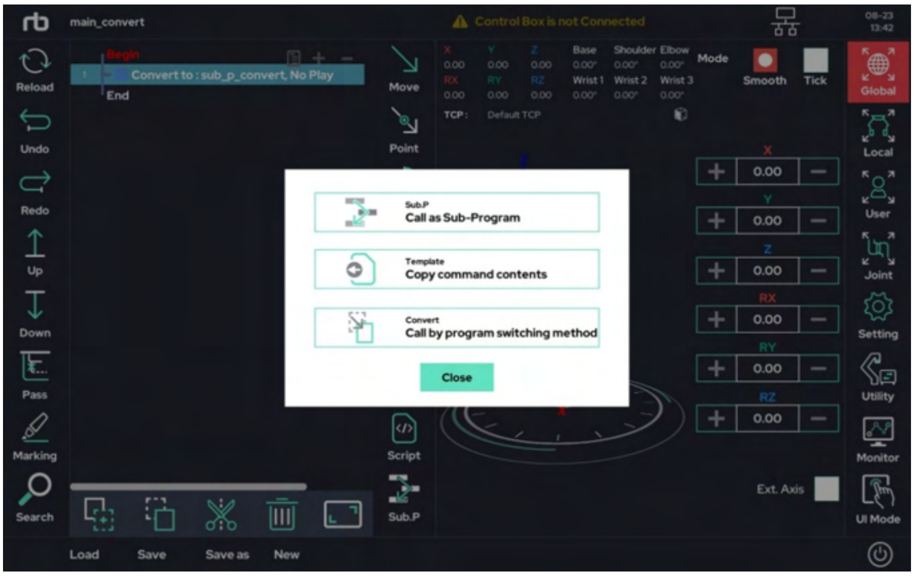
메인 프로그램에서 이벤트 쓰레드를 실행시키기 위한 기능입니다. 이벤트 쓰레드는 메인 프로그램에서 이벤트 쓰레드 호출이 실행되었을 때만 실행된다. 이때, 실행시킬 이벤트 쓰레드의 번호를 선택할 수 있다.

■ **프로그램 전환 방식 호출 :**

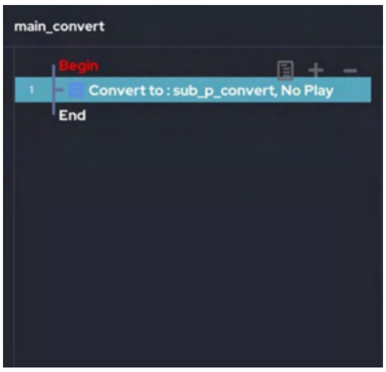


이 기능은 메인 프로그램을 전환하는 기능입니다. 기존의 서브루틴, 명령어 내용 복사와 다르게 메인 프로그램 자체를 바꾸는 기능이기 때문에 UI 에 나타나 있는 프로그램 또한 바뀌게 됩니다.

프로그램에서 Sub.P 아이콘을 클릭하면 아래와 같은 팝업창이 나오고 이때, 프로그램 전환 방식 호출을 누릅니다.



클릭한 후 나오는 파일 탐색기 팝업에서 전환하고자 하는 프로그램을 선택하면 아래와 같이 명령이 생성된다.



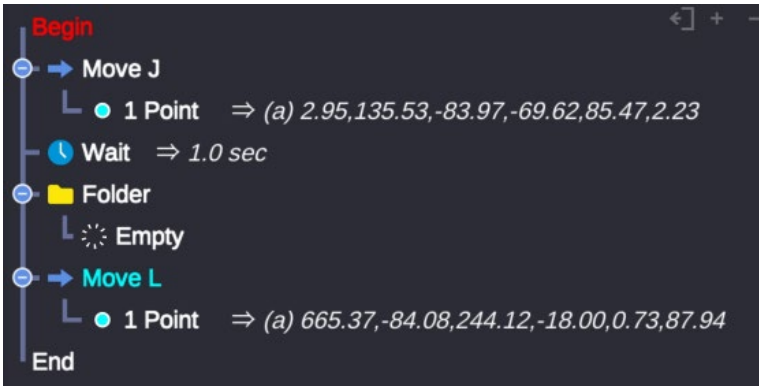
7.5 프로그램 편집하기

프로그램 티칭 리스트에 입력한 명령어의 순서나 구조를 변경할 수 있습니다.

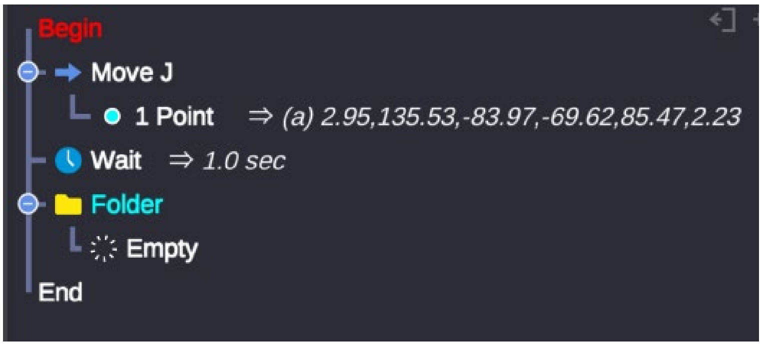
편집 아이콘의 설명은 6.1 을 참고하시기 바라며, 예제를 통해 프로그램 편집을 어떻게 할 수 있는지 설명합니다.

■ 잘라내기/ 붙여넣기 

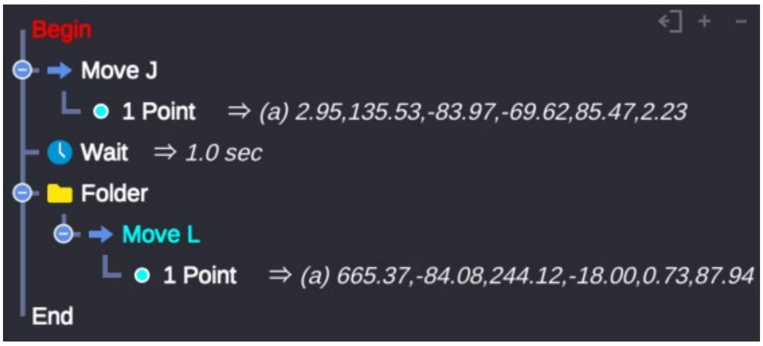
Step1) 잘라내고 싶은 명령어를 선택합니다. 선택된 명령어는 파란색으로 나타납니다. 예시에서는 MoveL 줄을 선택했습니다.



Step2) Cut (잘라내기) 버튼을 누릅니다. 잘라내기를 누르는 순간, 프로그램 Tree 에서 해당 줄이 사라집니다.

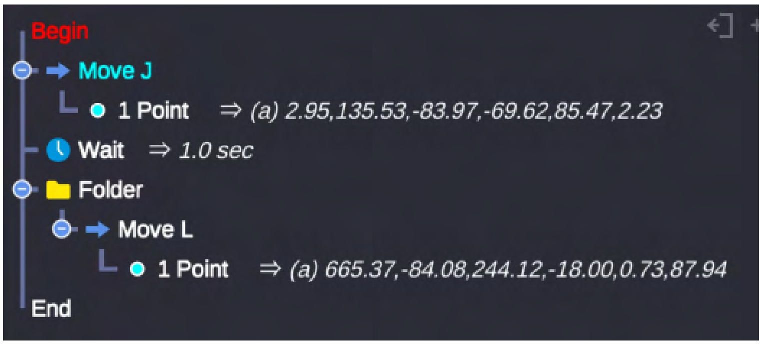


Step3) 붙어 놓고 싶은 위치를 클릭하고 붙여넣기(Paste) 버튼을 클릭합니다. 예시에서는 Folder 내부에 잘라낸 명령어를 다시 붙였습니다.



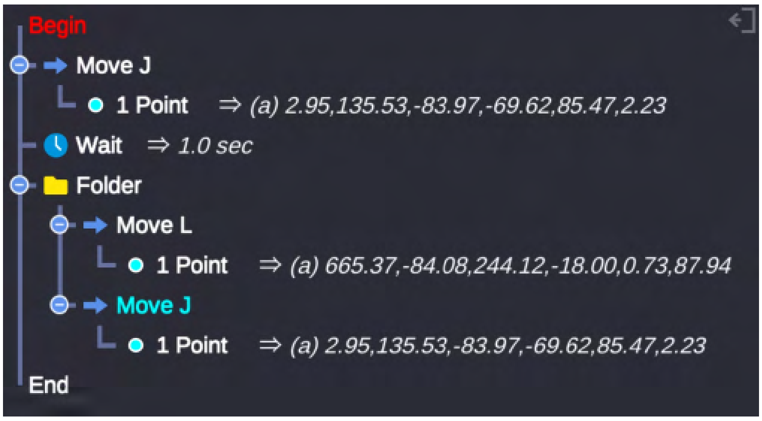
■ **복사하기/ 붙여넣기** 

Step1) 복사하고 싶은 선택합니다. 선택된 명령어는 파란색으로 나타납니다. 예시에서는 MoveJ 줄을 선택했습니다.



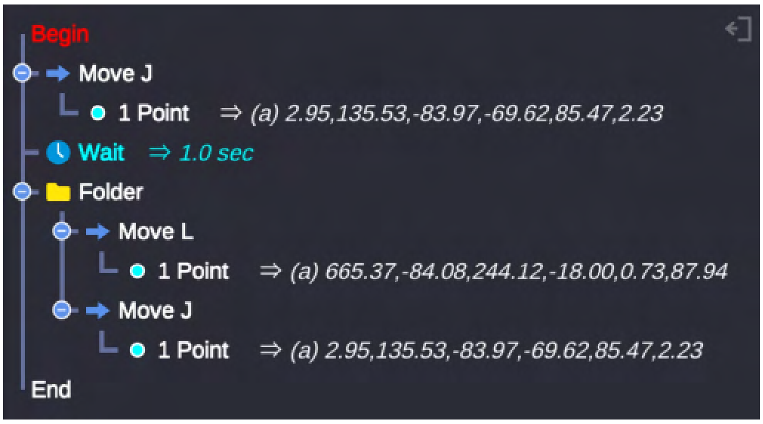
Step2) Copy(복사하기) 버튼을 누릅니다.

Step3) 붙어 놓고 싶은 위치를 클릭하고 붙여넣기 (Paste) 버튼을 클릭합니다. 예시에서는 Folder 하위의 MoveL 을 선택하여 그 아래에 복사된 명령어를 붙였습니다.

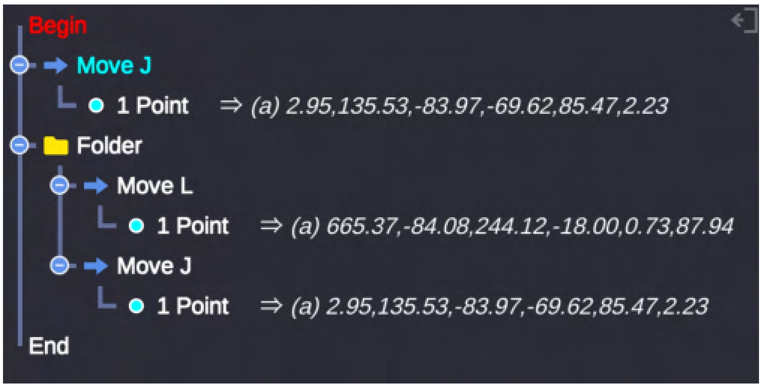


■ 삭제 

Step1) 삭제하고 싶은 명령어를 선택합니다. 선택된 명령어는 파란색으로 나타납니다. 본 예시에서는 Wait 명령어를 선택했습니다.

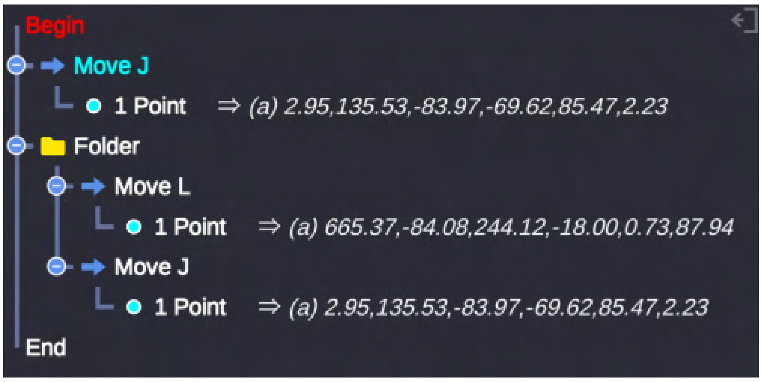


Step2) Del(삭제하기) 버튼을 클릭합니다. 아래와 같이 해당 명령어가 지워졌습니다.

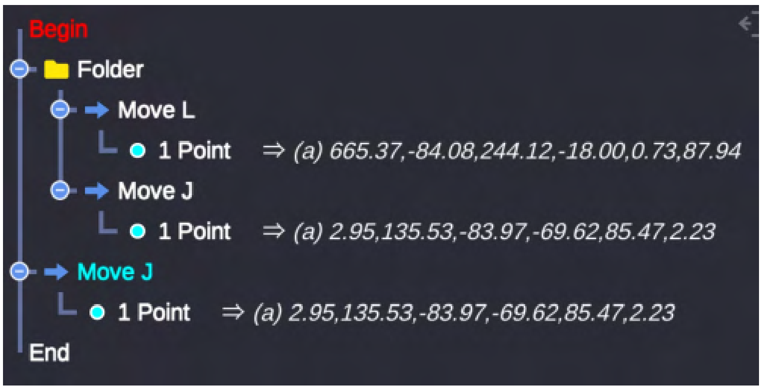


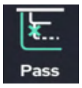
■ 이동 

Step1) 이동시키고 싶은 명령어를 선택합니다. 선택된 명령어는 파란색으로 나타납니다. 본 예시에서는 상단의 MoveJ 를 선택했습니다.



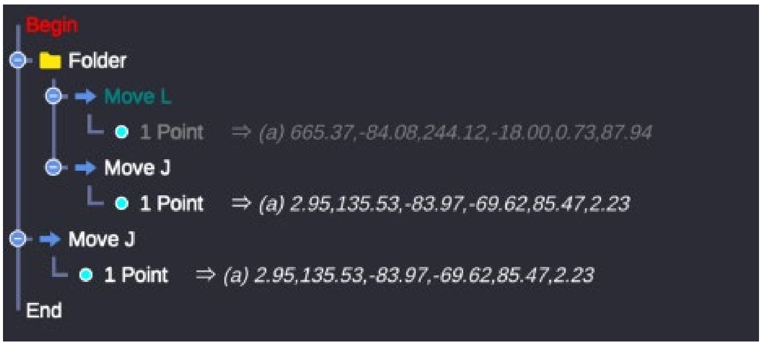
Step2) Down(내리기) 버튼을 클릭하여 보면 아래와 같이 선택했던 MoveJ 가 아래로 이동합니다.



■ 주석화 

Step1) 임시로 실행을 보류/차단하고 싶은 기능을 선택합니다. 선택된 명령어는 파란색으로 나타납니다. 본 예시에서는 MoveL 명령어를 선택했습니다.

Step2) Pass (주석화) 아이콘을 클릭합니다. 아래와 같이 해당 명령어가 어둡게 변합니다. 주석화 된 명령어는 프로그램 실행 중 작동하지 않습니다. 주석을 해제하려면 해당 명령어를 다시 선택하 Pass 버튼을 한 번 더 누르면 됩니다.



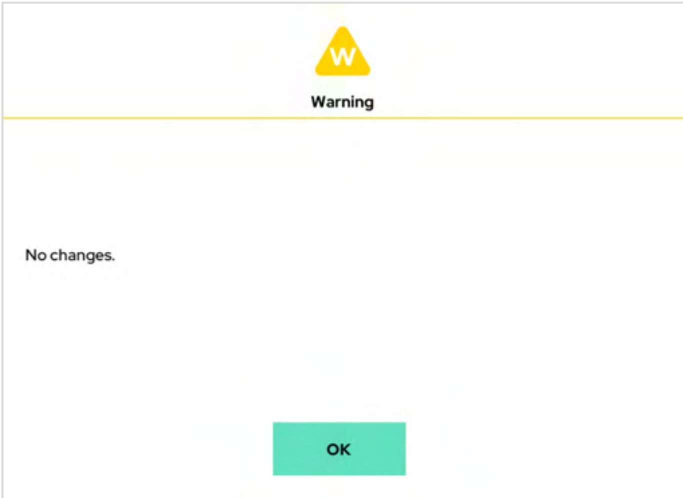
7.6 프로그램 관리하기

프로젝트를 저장, 불러오기, 다른 이름으로 저장 등을 할 수 있습니다.

- **프로젝트 저장**

현재 진행중인 프로젝트를 저장하기 위해서는 UI 작업 화면 하단 왼쪽편에 있는 

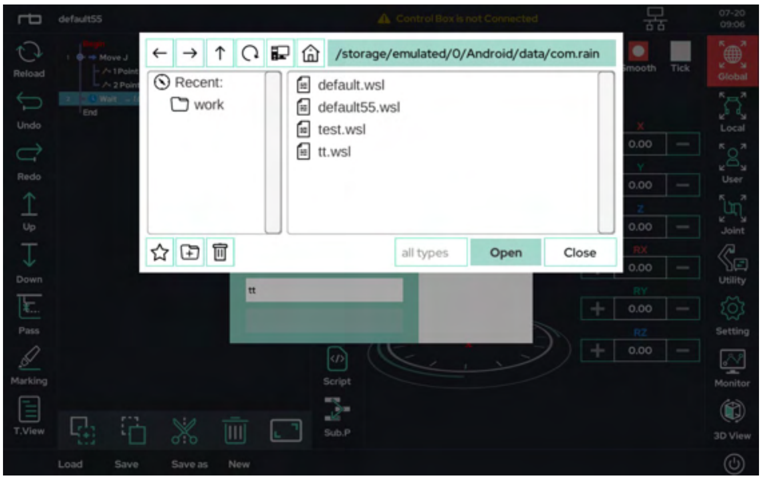
아이콘을 클릭하면 손쉽게 저장할 수 있습니다. 기존의 저장 내용과 변화가 없다면 아래와 같이 나타납니다.



■ 파일 불러오기

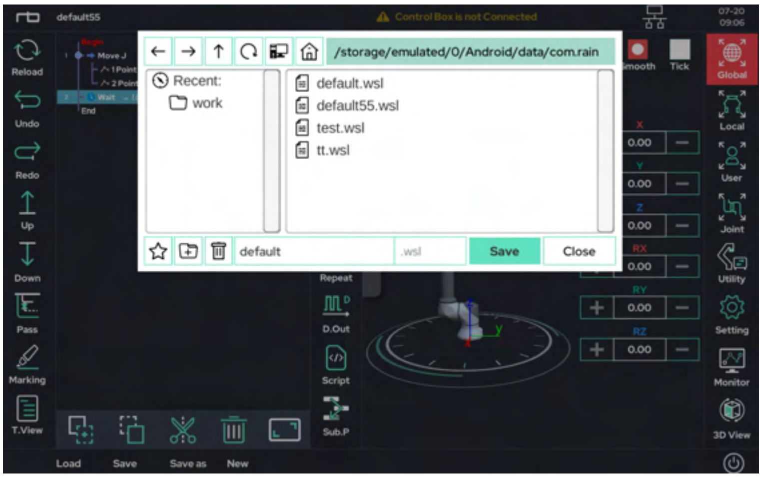
저장된 프로젝트를 불러오려면 아래 그림과 같이 작업 하단에 있는 FILE 을 클릭하면 선택할 수 있는 리스트가 나타납니다. 리스트 중 원하는 파일을 선택하면 해당 파일을 즉시 불러옵니다. 현재 프로젝트에 저장되지 않은 변경 사항이 있으면 사용자에게 저장하라는 메시지가 표시됩니다. 최근에 사용한 파일 만 나타납니다.

이전에 작업한 파일을 열고 싶을 경우, Other Program 을 클릭하여 파일 탐색기로 프로젝트를 찾아서 이전 프로그램을 불러들일 수 있습니다.



■ 다른 이름으로 저장하기

열려 있는 프로그램을 다른 이름으로 저장하여 관리할 경우, 작업 하단에 있는 FILE 을 클릭하여 나타나는 리스트 중 Save As 를 클릭하면 아래와 같은 팝업 창이 나타나고, 원하는 프로그램 이름으로 저장 가능합니다(default 라는 이름으로 저장은 불가능 함).



7.7 운용 유틸리티

Make 화면 우측에는 조그(Jog) 기능 이외에 시스템 운용을 위한 기타 유틸리티 기능들이 있습니다.

4) Utility: 자주 사용하는 자세 저장 기능, 시스템 입/출력 정보 보기 기능, 시스템 출력 테스트 기능 등, 주요 편의 기능들이 모여 있습니다.

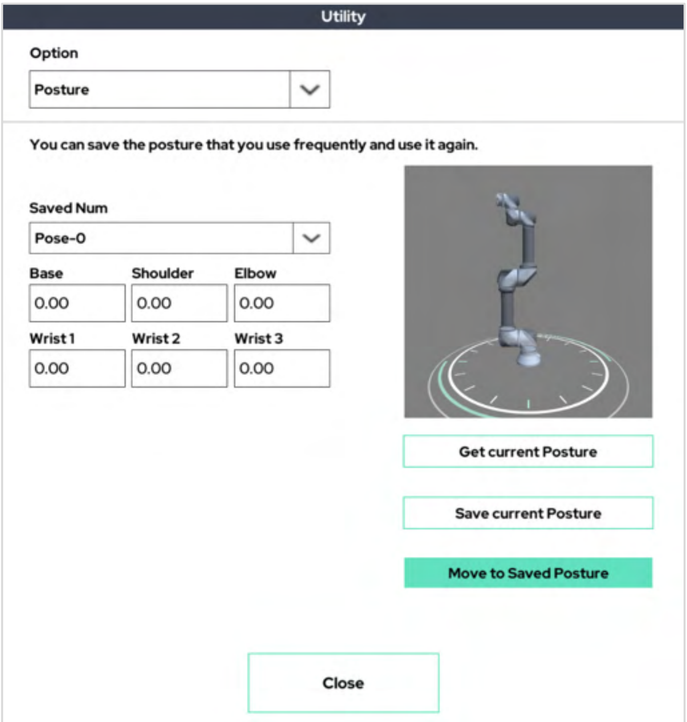
5) Setting: Setup 에서 설정하는 내용 중, 조그와 함께 사용하면 편리한 몇몇 설정 기능이 있습니다.

6) Monitor: 실시간 변수 모니터링 창 및 시스템 변수 모니터링 기능을 제공합니다.

7) UI Mode: 사용자의 수준, 사용자의 목적에 따라 UI 모드를 선택할 수 있다.

■ Utility 하위 기능

[Utility-Posture]



자주 사용하는 자세를 최대 20 개까지 UI 태블릿에 저장해 놓고 사용 가능합니다.

Get 버튼을 눌러서 현재 자세 정보를 가져오고, Set 버튼을 눌러서 저장합니다.

Move 버튼을 계속 누르고 있으면 저장된 자세로 이동합니다.

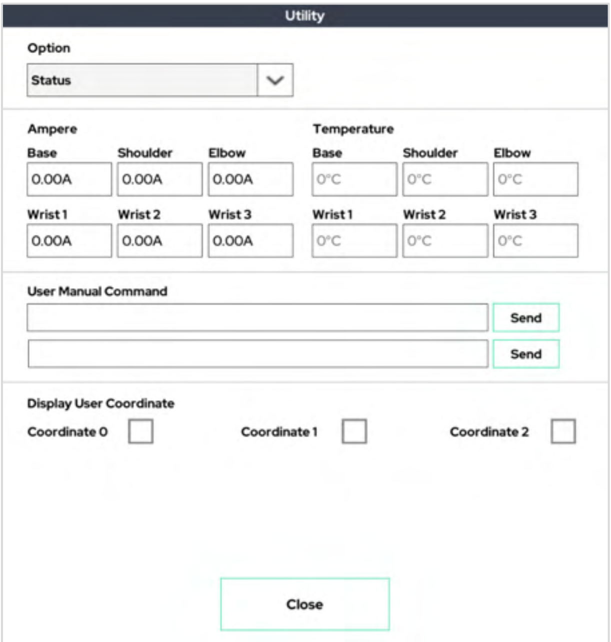
[Utility-Input Signal View]

컨트롤 박스와 툴 플랜지의 입력 신호 모니터링 창입니다.

[Utility-Output Signal View]

컨트롤 박스와 툴 플랜지의 출력 신호 모니터링 창입니다.

[Utility-Status]



로봇 팔의 온도 전류 모니터링 및 유저 좌표계 시각화 설정 창입니다.

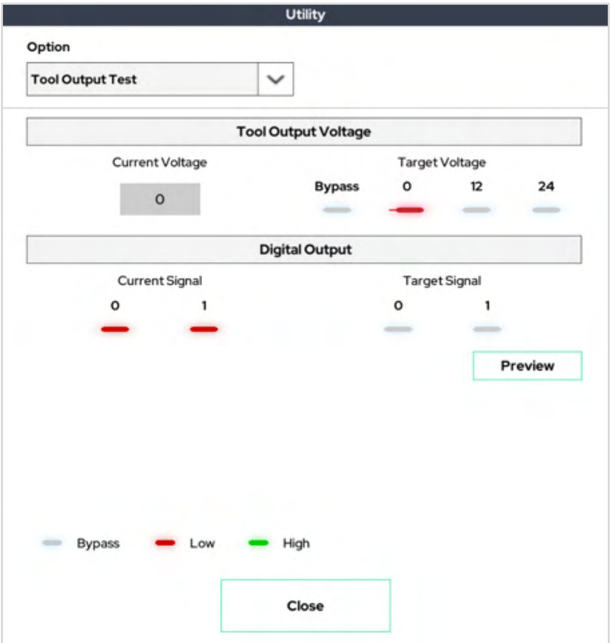
[Utility-Snap]

직접 교시 모드 사용시 적용될 스냅(snap) 모드 선택 창입니다.

[Utility-Box Output Test]

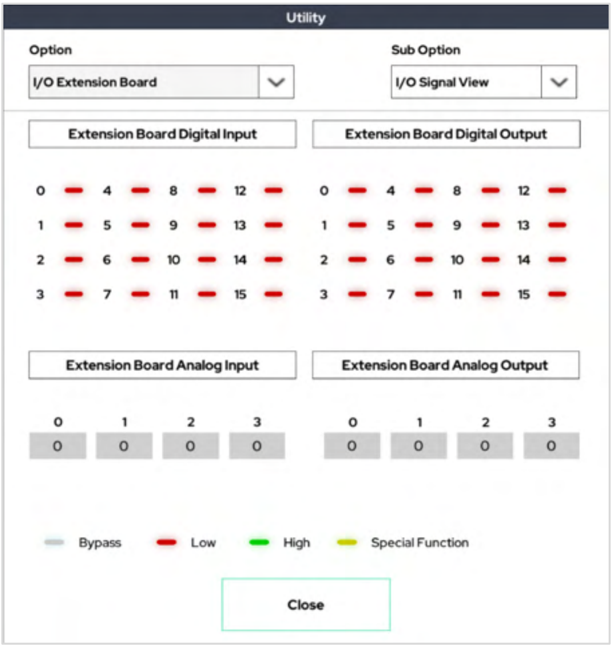
컨트롤 박스의 출력을 테스트할 수 있는 창입니다.

[Utility-Tool Output Test]

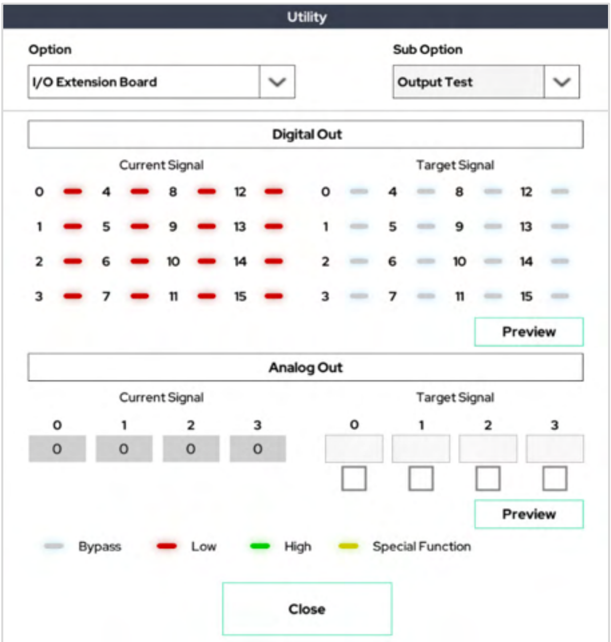


툴 플랜지의 출력을 테스트할 수 있는 창입니다.

[Utility-I/O Extension Board]



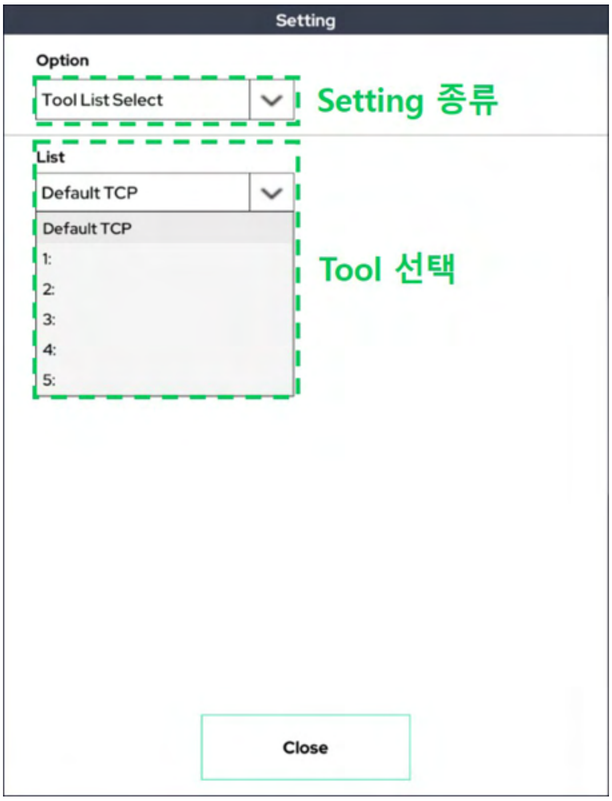
I/O 확장 모듈의 입출력 신호 모니터링 창입니다.



I/O 확장 모듈의 출력을 테스트할 수 있는 창입니다.

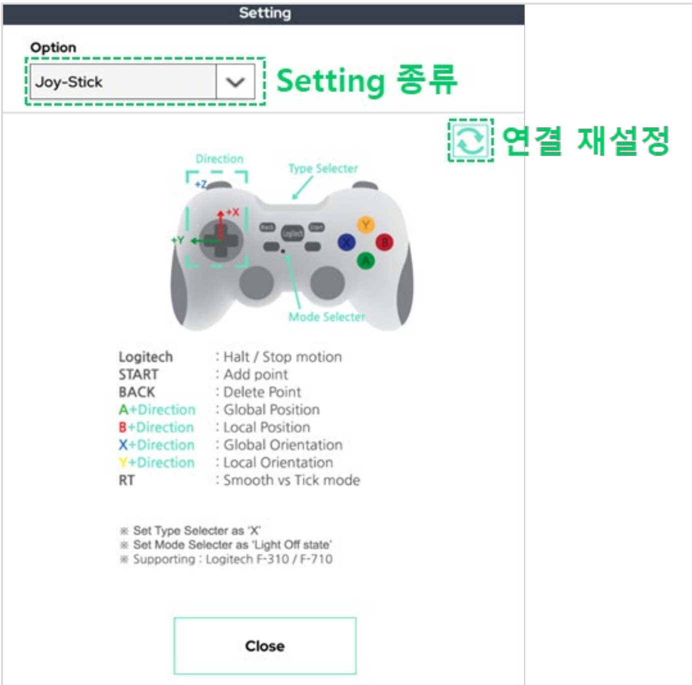
■ Setting 하위 기능

[Setting-Tool List Select]



미리 저장한 TCP 목록에서 사용할 TCP 를 설정하는 Tool List Select 설정 기능이 있습니다.

[Setting-Joystick]



유/무선 조이스틱을 활용하여 로봇을 티칭 시, 조이스틱 연결 설정 기능이 있습니다.

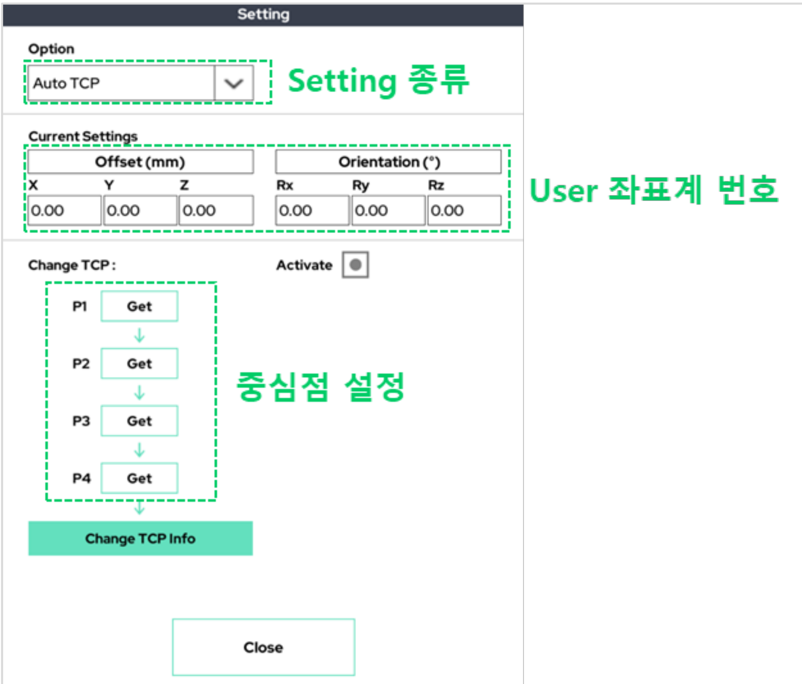
[Setting-User Coordinate]

사용자 정의 좌표계를 세팅합니다. 3 점 설정 모드입니다(Setup-Coordinate 참고).

[Setting-User Coordinate Center]

사용자 정의 좌표계의 회전 값은 유지하고, 중심점(offset)만 설정합니다.

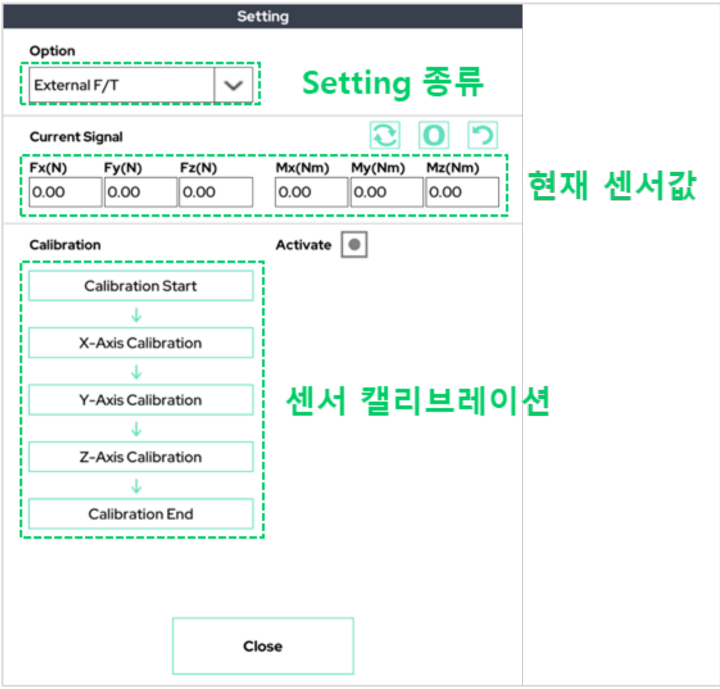
[Setting-Auto TCP]



자동으로 로봇 말단(TCP)의 위치를 찾아주는 기능입니다.

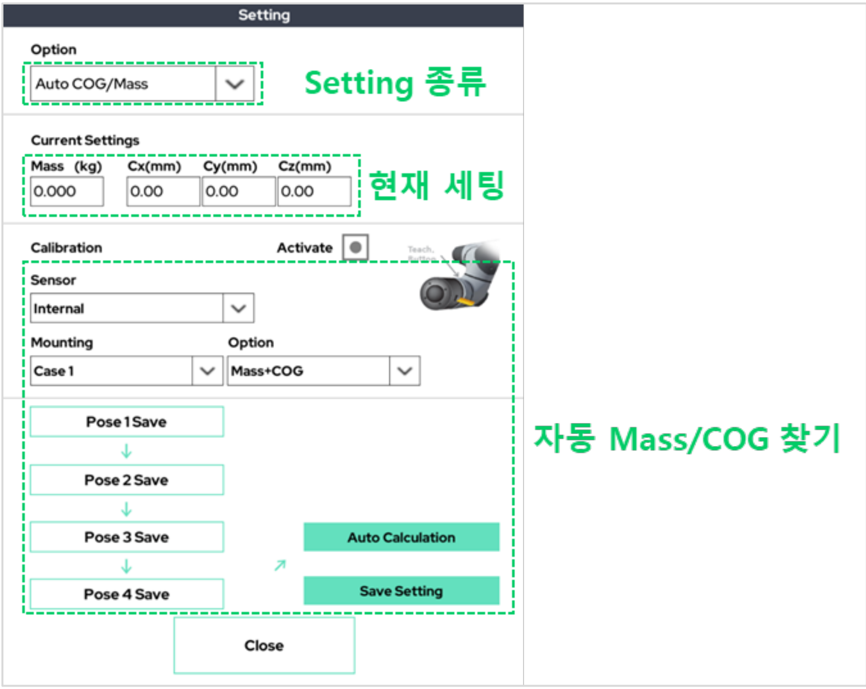
설정하고자 하는 TCP 포인트를 3 차원 공간상에 같은 지점으로 유지하는 4 개의 다른 자세를 입력하면 자동으로 TCP 위치 값을 계산합니다.

[Setting-External F/T]



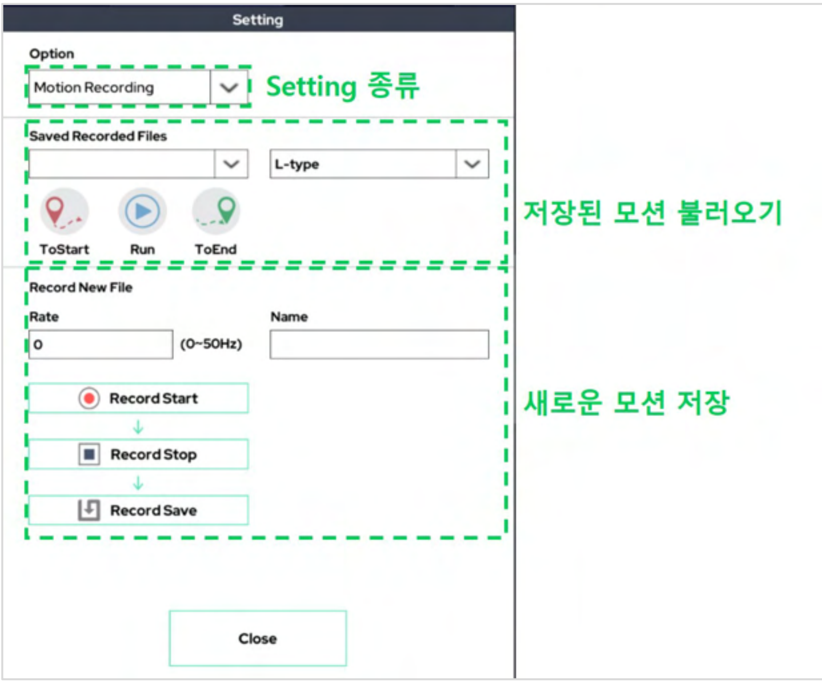
외부 F/T 센서 (예: Robotiq F/T 센서) 값을 확인하고 캘리브레이션 할 수 있는 창입니다.

[Setting-Auto COG / Mass]



내부 / 외부 F/T 센서를 사용해서 툴에 부착된 무게와 무게 중심을 찾아주는 기능입니다.

[Setting- Motion Recording]



직접 교시 (중력 보상) 기능을 통해 동작을 녹화하는 기능입니다. 이 기능을 통해서 녹화된 동작은 Replay 기능을 통해서 프로그램에서 사용 가능합니다.

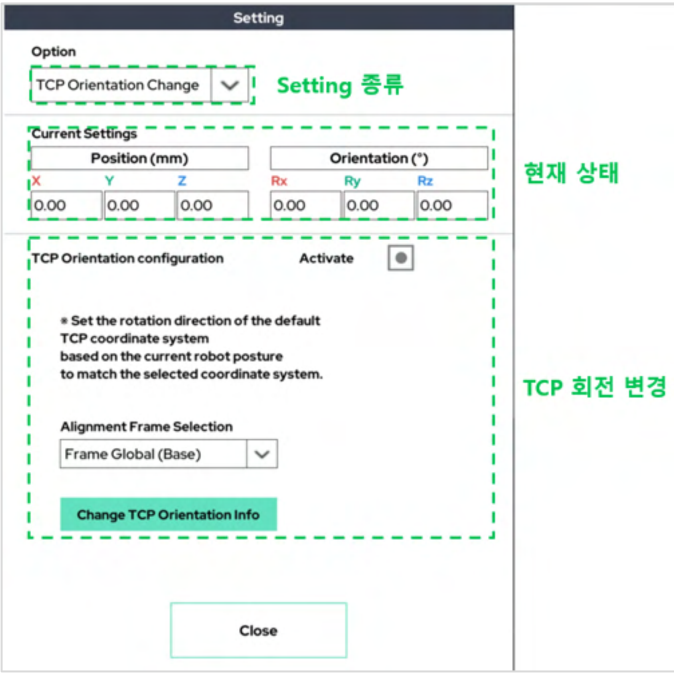
[Setting-I/O Logging]

하나의 디지털 입출력을 정해 그 입출력의 값의 변화를 기록하고 이를 그래프로 보여주는 기능입니다.

[Setting-Welder Wire Control]

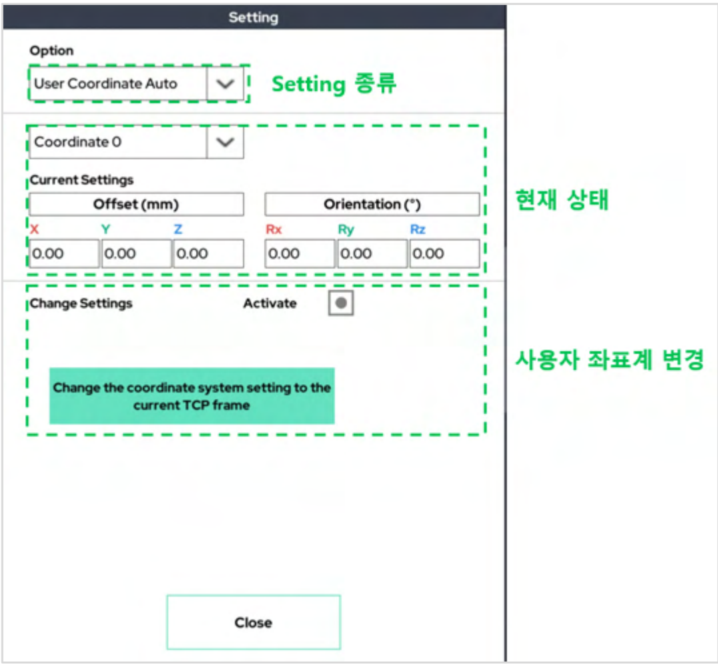
용접기의 와이어를 제어할 수 있는 기능입니다.

[Setting-TCP Orientation Change]



선택한 좌표계와 일치하도록 현재 로봇 자세를 기반으로 기본 TCP 좌표계의 회전 방향을 설정합니다.

[Setting-User Coordinate Auto]

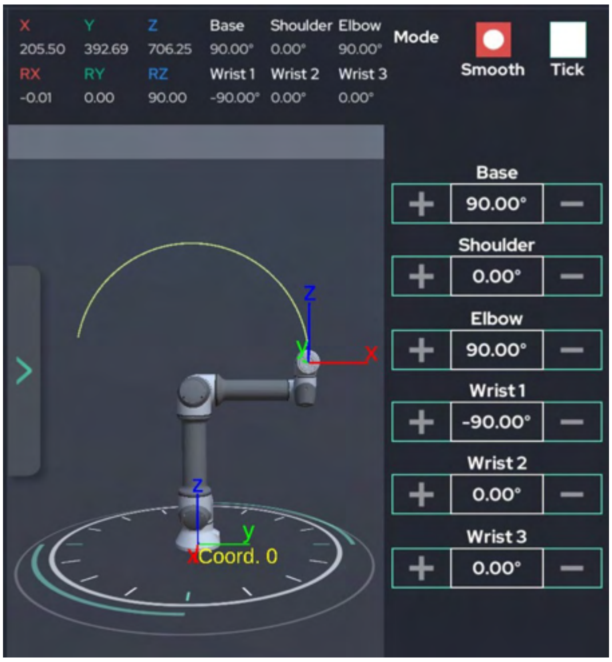


로봇의 현재 TCP 좌표계를 사용자 좌표계로 설정합니다.

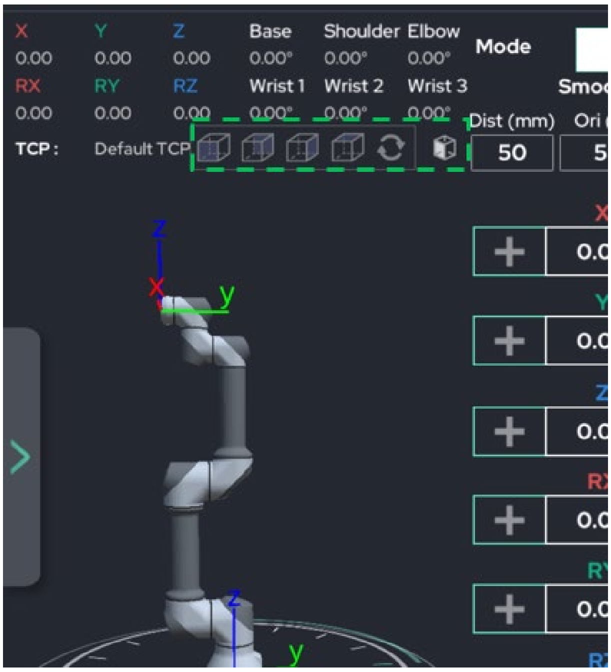
■ Monitor 기능

6 단원의 Monitor 명령어와 함께 사용하는 기능으로써, 시스템 기본 변수 및 사용자가 설정한 변수들의 값을 실시간으로 관찰할 수 있는 창입니다.

우측 상단의 녹화 기능을 켜면, 3 차원 뷰어에서 로봇 툴의 TCP 궤적이 3 차원 뷰어에 녹화가 됩니다. (노란 실선)



■ 3D View 기능



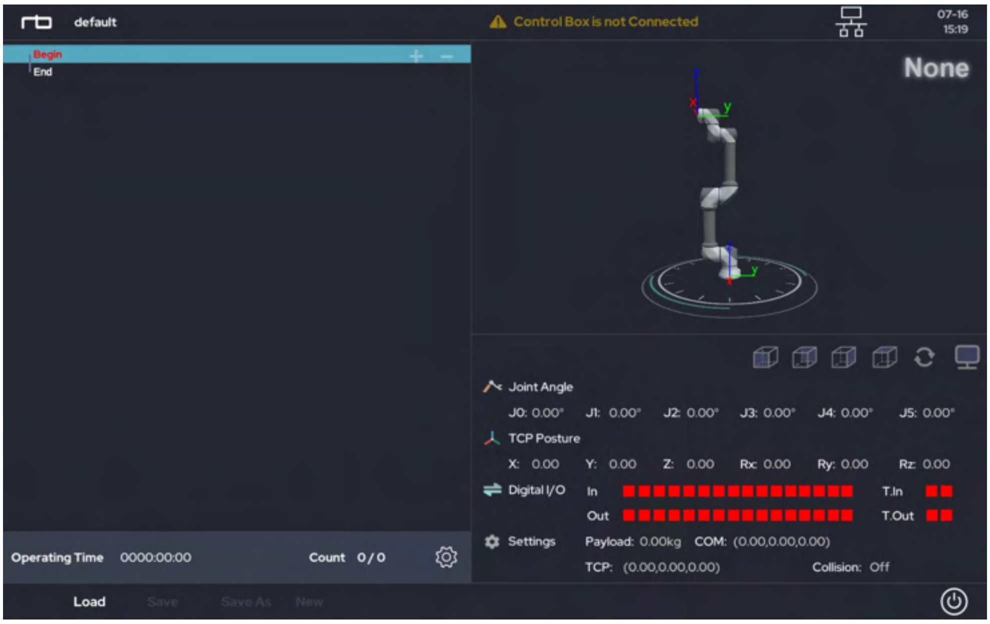
사용 빈도가 높은 3 차원 시야 각도가 지정되어 있습니다. 3 차원 뷰어의 시각을 빠르게 바꾸고 싶은 경우 사용 가능합니다.

제 8 장 로봇 시작하기

8.1 로봇 구동하기

로봇을 구동하기 위해선 실행 화면(Play)로 이동하십시오.

- 실행 화면은 다음과 같습니다.



- 로봇의 연결 상태를 확인하시기 바랍니다. 로봇이 연결되어 있지 않다면 6.2 를 참고하셔서  아이콘을 이용하여 로봇과 연결하십시오. 이 때의 주의사항에 유의하시기 바랍니다.
- 저장된 프로젝트를 불러옵니다. 프로젝트 불러오기는 6.6 을 참고하시기 바랍니다.
- 모든 셋팅이 끝났다면 화면 하단 부분에 있는 PLAY 버튼을 누릅니다.
- 로봇의 상태가 Initial position 에 위치해 있지 않을 경우 안내창이 뜹니다. approach 버튼을 눌러 로봇과 Initial position 을 일치시킵니다.
- 기본적으로 실행 화면에서 로봇을 구동하게 되면 실행하고 있는 프로젝트는 무한으로 반복하게 됩니다. 상단의 Count 버튼을 눌러 반복 횟수를 지정할 수 있습니다.
- 작업이 실행 중에도 속도조절이 가능합니다.

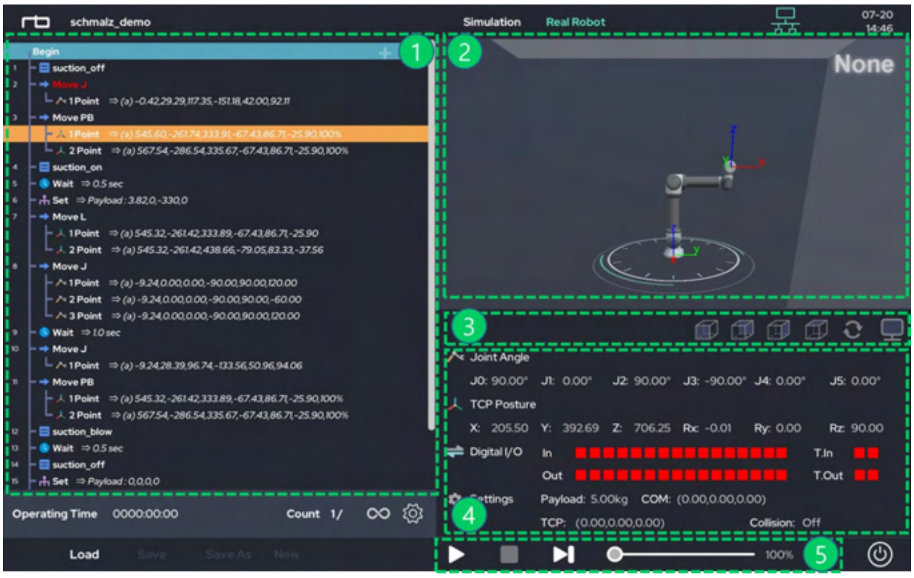
경고

경고:

1) 사용자는 로봇 팔을 구동하기 전에 로봇 시스템에 대한 위험성 평가를 충분히 수행한 후, 로봇에 적용 가능한 안전 기능을 모두 적용하고 로봇 팔을 구동해야 합니다.
2) 초기화 과정 중 로봇에 문제가 있거나, 로봇 설치 각도, 부하 설정 등이 부정확할 경우 로봇 초기화에 실패할 수 있습니다.
3) Play 화면에서의 실행은 Simulation 없이 실제 로봇이 움직입니다. 사용법을 충분히 숙지한 후 사용하기 바랍니다.
4) 구동 중에는 Make 나 Setup 과 같은 다른 화면으로 이동이 불가능 합니다. 화면 전환을 위해서는 동작 중인 프로그램을 완전히 정지시킨 이후 사용하시기 바랍니다.
5) 구동 중 태블릿 UI 를 뽑아도 무방합니다.

8.2 로봇 상태 확인

실행화면에서 현재 구동 중인 로봇 팔의 상태를 확인할 수 있습니다.

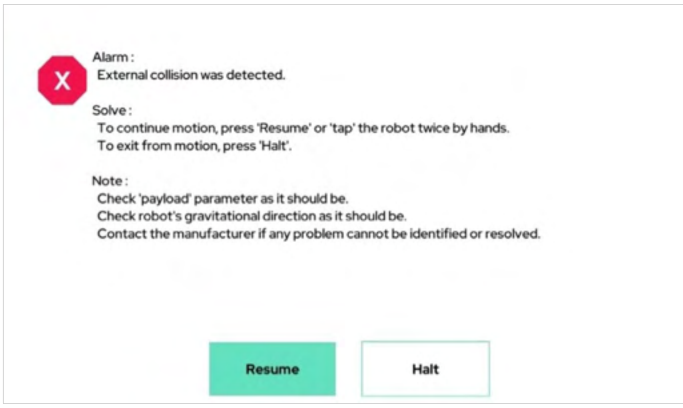


번호	설명
①	프로그램 흐름 뷰어입니다.
②	3 차원 뷰어입니다.
③	3 차원 뷰어의 시각을 조절합니다.
④	시스템 상태 정보, 시스템 변수 정보 등을 보여줍니다.
⑤	재생 / 정지 / 속도 조절 바가 있습니다.

8.3 로봇 운용 중 발생한 특이사항 대처방법

로봇 운용 중에는 여러가지 문제 상황이 발생할 수 있고 그에 따른 대처법들이 있습니다.

1. 외부 충돌(Out Collision)



로봇이 작동 중에 외부의 물체와 충돌했을 때 로봇이 작업을 일시정지 합니다.

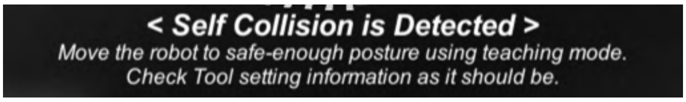
우선 외부 충돌 요인을 제거합니다. 이후 두가지의 선택지가 있습니다. 로봇이 하던 작업을 중단하고 정비를 하려면 Halt 버튼을 눌러 작업을 종료합니다. 로봇이 하던 작업을 계속하기 위해서는 Resume 버튼을 눌러 작업을 재개합니다.

TOK TOK 회복법

외부 충돌로 인한 일시 정지 시 로봇 팔의 말단부를 가볍게 2 회 두드리면 일시정지 상태가 회복됩니다(Resume 버튼과 동일 효과).

---

2. 자가 충돌 (Self Collision)



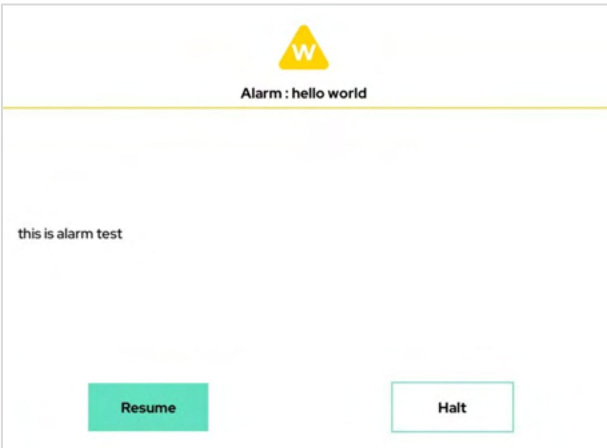
로봇의 일부분이 지정된 작업 범위를 벗어나거나 로봇 자체적으로 충돌 위험이 있는 모션에 근접했을 때 로봇이 작업을 정지합니다.

티칭 버튼을 사용하여 직접 교시 기능을 통해 로봇의 상태를 정상상태로 만들어야 합니다. 이후 티칭한 모션 및 작업 범위 설정을 검토합니다.

Make 화면의 Simulation 모드에서 발생시 다음과 같은 방법으로 복구할 수 있습니다.

- 로봇의 모션 관련 버튼을 통해 로봇을 자가충돌상황에서 회복
- Real 모드로 일시적으로 변경해 실제 로봇의 관절 정보를 받아 회복
- 티칭 버튼을 눌러 실제 로봇의 관절 정보를 받아 회복

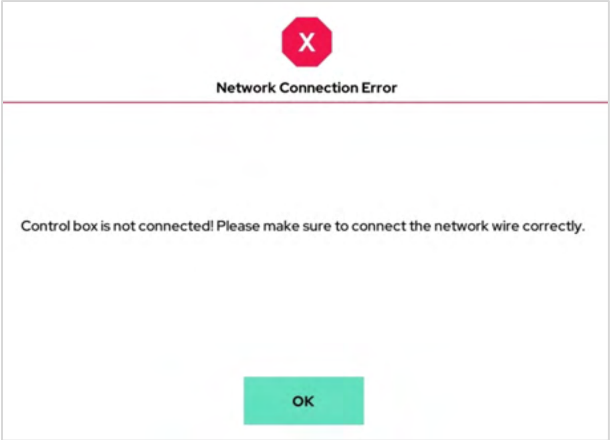
3. 알람 메시지



프로그램이 진행되는 중 Alarm 메시지 기능이 실행될 때 로봇이 작업을 일시 정지합니다.

Resume 버튼을 눌러 작업을 진행하거나, Halt 버튼을 눌러 작업을 정지시킵니다.

4. 태블릿 PC 분리



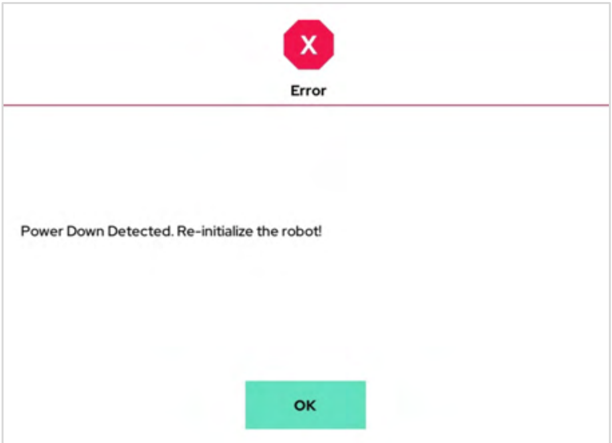
컨트롤 박스(컨트롤러)와 태블릿 PC 가 분리될 때 발생합니다.

실수로 USB 케이블이 분리되었다면 케이블을 다시 연결한 후 안내창에 따라 태블릿 PC 를 컨트롤 박스에 다시 연결합니다.

케이블이 손상된 경우라면 케이블을 교체 후 안내창에 따라 로봇을 다시 연결합니다.

연결이 지속적으로 이루어 지지 않는다면, 다른 USB 포트로 시도해 보세요.

5. 외부 전력공급 중단



로봇 팔에 공급되는 전력이 중단될 때 발생합니다. 강제로 로봇 컨트롤 박스의 AC 단을 끊거나, 비상 정지 스위치를 누를 경우 나타납니다.

컨트롤 박스의 전력을 복구하거나 비상 정지 스위치를 정상적으로 복구한 후, 다시 로봇을 초기화 한 이후 사용합니다.

6. 관절 제어기 에러 검출

관절제어기에서 로봇을 제어하기 어렵다고 판단되는 경우에 로봇을 정지합니다.

다음과 같은 항목이 존재합니다.

Big Error	입력 레퍼런스와 엔코더의 차이가 특정 값 이상 벌어졌을 때 발생
Jam Error	엔코더 값이 변화가 없으나 특정 값 이상의 전류가 특정 시간 이상 흐를 때 발생
Overcurrent Error	특정 값 이상의 전류가 특정시간 이상 흐를 때 발생
Temperature Error	제어보드의 온도가 특정 값 이상일 때 발생
Mode Error	메인 제어기에서 명령한 제어모드가 현재 관절제어기의 모드와 일치하지 않을 때 발생

제 9 장 환경 설정하기

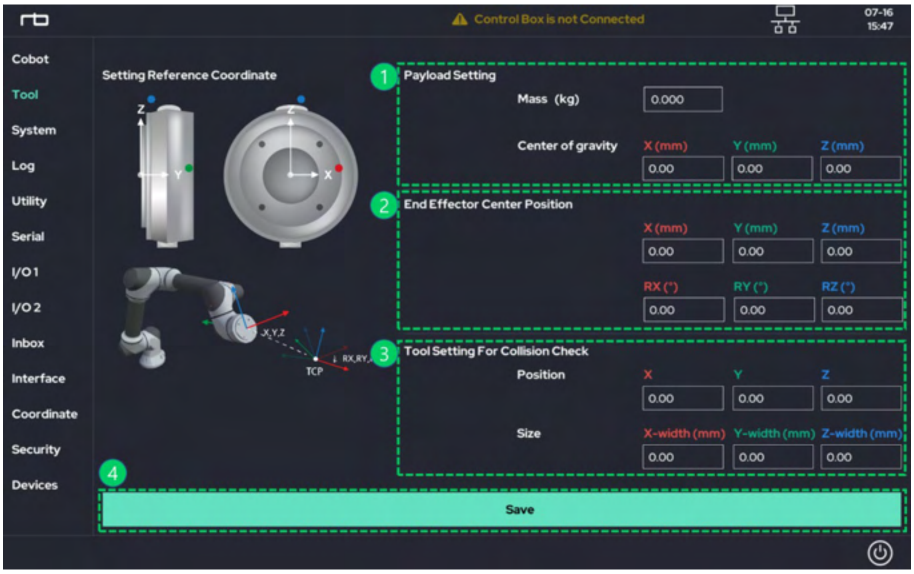
9.1 SET-UP(COBOT)

로봇 팔에 관한 기본 환경을 설정할 수 있습니다

①	충돌 감도를 설정합니다. 감도가 낮을수록 약한 충격에도 로봇이 멈춥니다. 충돌 감지 기능 자체를 on/off 가능합니다.
②	로봇의 활동 영역을 설정합니다. Enable Work Space 를 설정하면 3D 상에 활동 한계 영역이 표시됩니다. 활동 영역을 벗어나면 자가 충돌로 인식하여 로봇이 멈추게 됩니다. 본 기능은 on/off 가능합니다.
③	현재 운용중인 UI 로봇 모델과, 연결된 컨트롤 박스의 로봇 모델을 표기합니다.
④	충돌 후 정지 모드를 선택합니다. 제자리 정지(General Stop)과 회피 정지 (Evasion Stop) 중 선택합니다.
⑤	로봇의 설치 각도를 설정합니다. 제시된 예시 사진 중 해당하는 설치 각도가 있으면 그림을 클릭합니다. 직접 입력하려면 Global 좌표계를 기준으로 중력가속도의 방향벡터를 입력합니다.
⑥	변경된 세팅을 저장합니다.

9.2 SET-UP(TOOL)

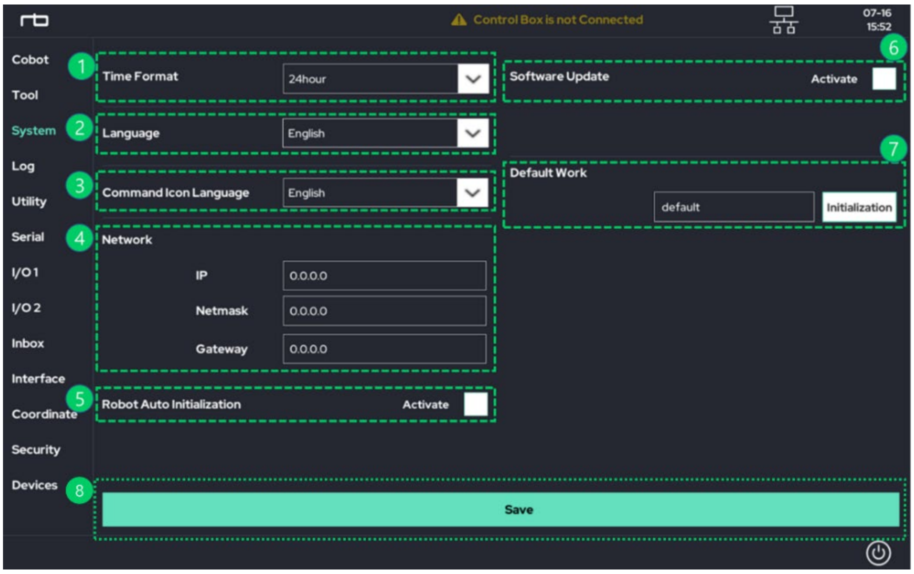
설치된 툴에 관한 설정을 할 수 있습니다.



①	자가 충돌을 방지하기 위해 툴 단에 장착되는 가상의 박스 정보를 입력합니다. 박스의 사이즈와 박스 중심점의 위치를 입력합니다. 위의 세팅은 툴 플랜지에 장착된 제조사 기본 좌표계를 기준으로 합니다.
②	로봇에 걸리는 하중에 관한 설정을 합니다. 질량은 'Mass'란에(단위 kg) 로봇 모델 별 가반하중에 따라 입력 범주가 다릅니다. 질량 중심까지의 거리는 'Center of gravity'란에(단위 mm) 입력합니다.
③	로봇 팔 말단 (TCP)의 위치와 좌표계 방향을 설정합니다. 말단의 좌표계는 제조사의 기본 말단 좌표계에서 오일러 각으로 얼마나 회전시킬 지를 설정합니다. 위의 세팅은 툴 플랜지에 장착된 제조사 기본 좌표계를 기준으로 합니다.
④	변경된 세팅을 저장합니다.

9.3 SET-UP(SYSTEM)

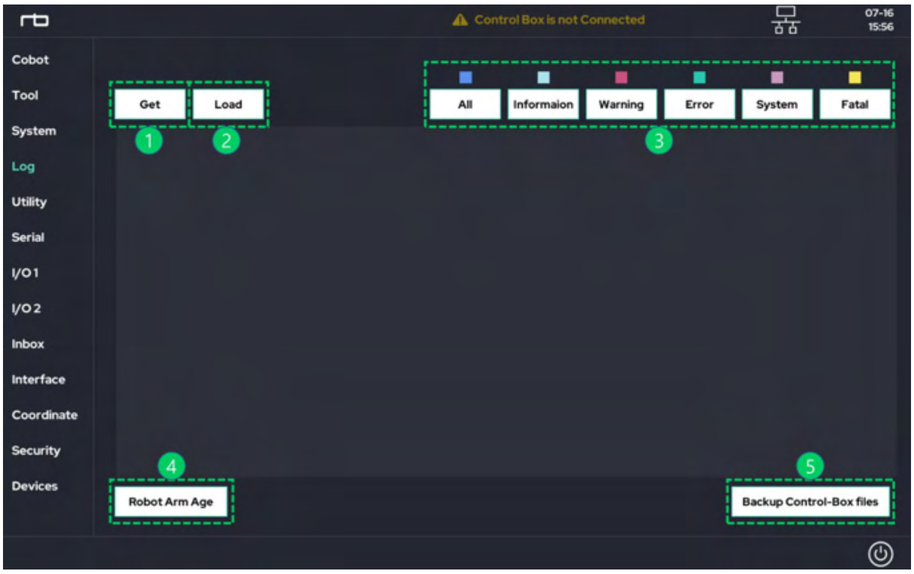
표시 단위, 날짜 및 시간, UI 암호, 시스템 업데이트 등을 설정할 수 있습니다.



①	UI 프로그램의 날짜 표기법을 설정합니다.
②	언어를 설정합니다.
③	명령 아이콘 언어를 설정합니다.
④	컨트롤 박스의 IP 정보를 세팅합니다. 이곳에서 세팅한 IP 는 Socket 기능, Modbus 기능, 외부 스크립트 제어 기능 등에 영향을 미칩니다.
⑤	로봇 팔 자동 활성화 기능입니다. 이 기능을 사용하면, 로봇 팔은 활성화가 가능한 상태가 되면 자동으로 활성화를 진행합니다. 자동 활성화된 로봇 팔은 기본 Real-Mode 입니다. 이 기능을 사용하기 위해서는 컨트롤 박스 디지털 입력에 Auto-Initialization Key 기능이 정의되어야 하고, 해당 key 에 입력이 있을 경우 작동합니다(이중화).
⑥	컨트롤 박스 소프트웨어 업데이트를 위한 기능입니다. 부록(소프트웨어 업데이트)를 참고하세요.
⑦	컨트롤 박스 및 UI 가 초기 실행 시 기본으로 열릴(load) 프로젝트를 설정합니다.
⑧	변경된 세팅을 저장합니다.

9.4 SET-UP(LOG)

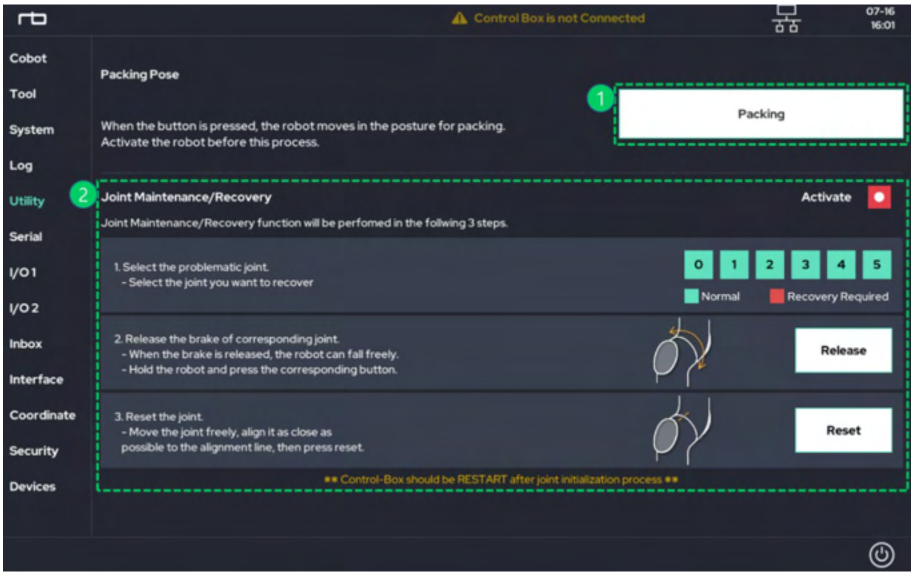
로봇 팔의 시스템 로그 상태를 확인할 수 있습니다.



①	컨트롤 박스로부터 로그 파일을 태블릿 PC 로 복사합니다. 복사할 파일 크기에 따라 몇 초 소모됩니다.
②	1 의 과정을 통해 복사해온 로그 파일을 여는 기능입니다.
③	로그를 필터를 통해 종류 별로 보는 기능입니다. I : Information W : Warning E : Error S : System F : Fatal
④	로봇 팔의 상태 정보를 확인하는 기능입니다.
⑤	제어 박스에 저장된 프로그램 파일 / 로그 파일 / 세팅 파일 등을 태블릿 PC 로 복사해오는 기능입니다. 복사(백업)된 파일들은 태블릿 PC 의 특정 경로에 저장됩니다.

9.5 SET-UP(UTILITY)

로봇의 포장 및 비상 복구에 관한 기능을 제공합니다.



①	로봇을 포장할 때의 자세를 만들 수 있습니다. 이 기능을 사용하기에 앞서 로봇 팔은 활성화 되어있어야 합니다. 이 기능을 사용하기에 앞서 로봇 팔에 부착된 모든 부착물을 제거하세요. 버튼을 누르고 있는 동안 로봇 팔은 포장 자세로 이동합니다. 포장 자세에 도달하면, 포장 자세에 도달하였다는 알람이 뜹니다.
②	축의 기준점을 공장 초기화 상태로 세팅할 수 있습니다. 이 기능은 로봇이 비정상 작동 시 복구를 위한 기능이므로 주의해서 사용해야 합니다. Step1. 리셋 할 축을 정합니다 Step2. Release 버튼을 눌러 브레이크를 풀어 자유로운 상태로 만듭니다. Step3. 축에 표시된 홈을 정렬하고 리셋 버튼을 눌러 축을 초기화 합니다.

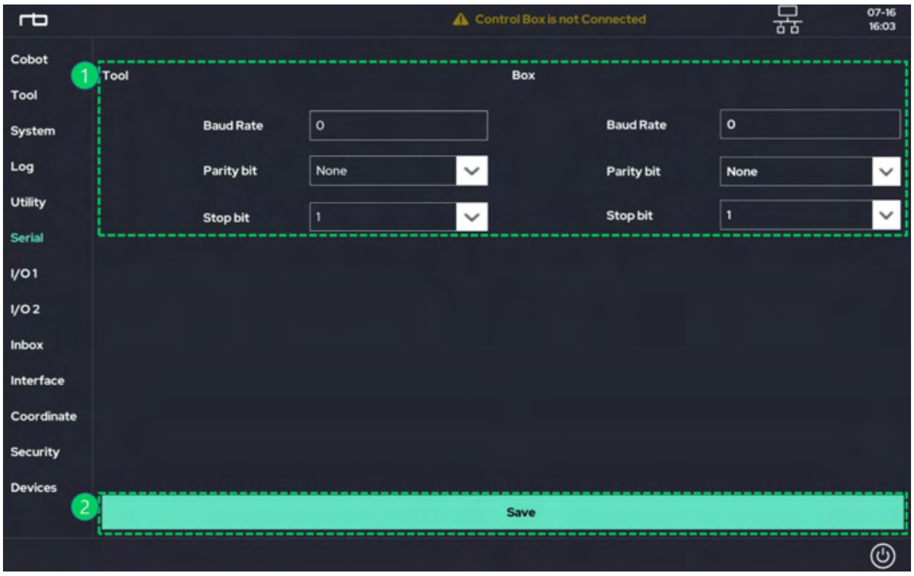
주의

주의:

1) 비상 관절 복구 기능(Emergency Joint Recovery) 모드 사용 전 본 기능의 특성과 작동 법을 정확히 숙지 후 사용하세요.
2) 로봇 운반 / 이송 시 제조사에서 제공한 포장 박스에 포장 자세로 넣어야 합니다.

9.6 SET-UP(SERIAL)

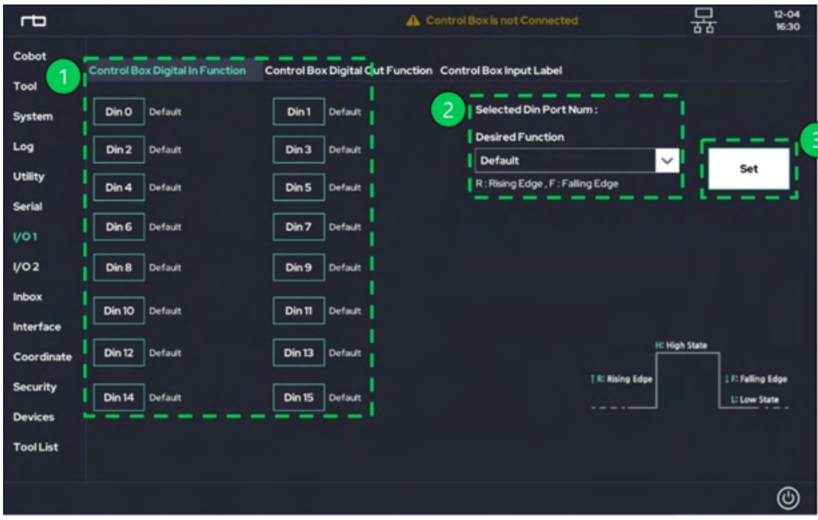
로봇의 툴과 컨트롤 박스와의 Serial 통신을 설정할 수 있는 곳입니다.



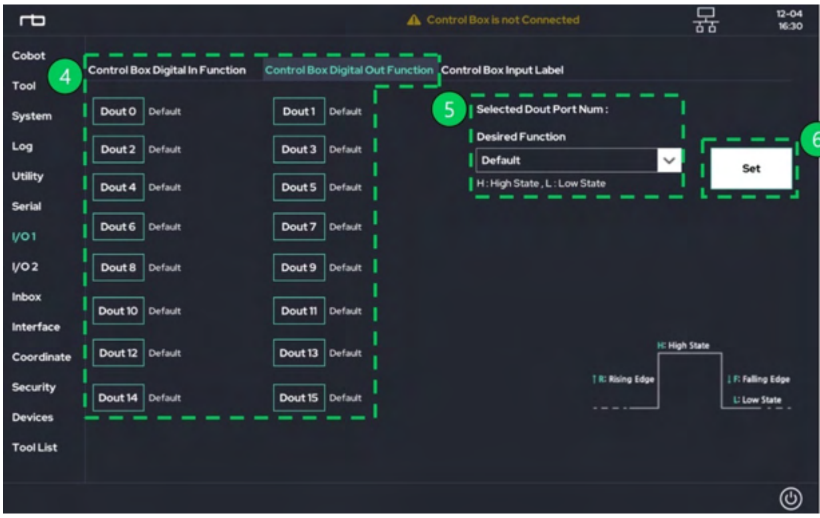
①	RS485 통신 옵션을 정합니다. 통신 속도(Baud Rate), 정지 비트(Stop bit), 패러티 비트(Parity bit)를 설정합니다.
②	변경된 세팅을 저장합니다.

9.7 SET-UP(I/O 1)

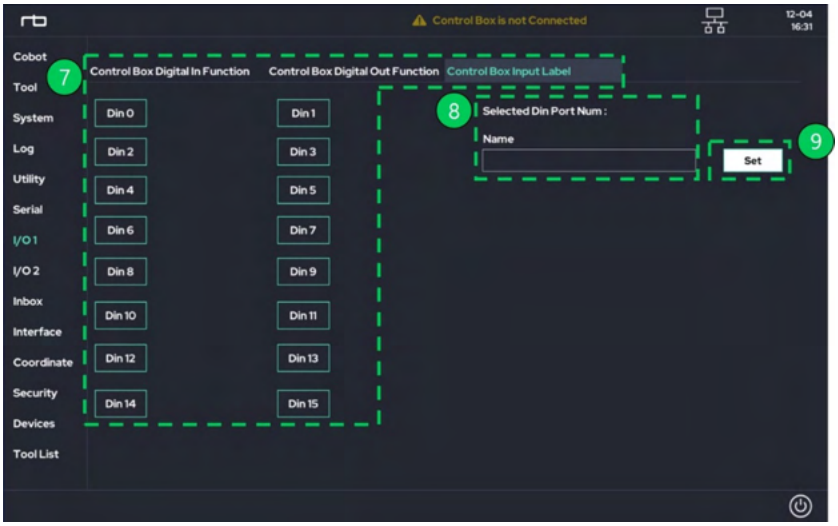
컨트롤 박스의 GPIO 포트의 기능을 설정할 수 있는 곳입니다.



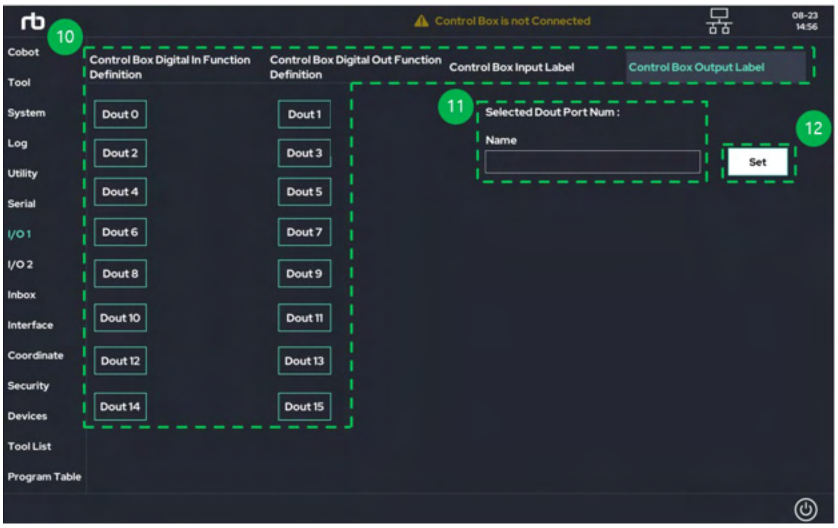
①	기능을 변경할 입력 포트를 선택합니다..
②	어떤 기능으로 변경할지 선택합니다. 각 기능 옵션에 대한 설명은 다음 페이지에 있습니다.
③	선택한 기능으로 저장합니다.



④	기능을 변경할 출력 포트를 선택합니다..
⑤	어떤 기능으로 변경할지 선택합니다. 각 기능 옵션에 대한 설명은 다음 페이지에 있습니다.
⑥	선택한 기능으로 저장합니다.



⑦	이름을 설정할 입력 포트를 선택합니다.
⑧	사용할 이름을 입력합니다.
⑨	선택한 이름으로 저장합니다.



⑩	이름을 설정할 출력 포트를 선택합니다.
⑪	사용할 이름을 입력합니다.
⑫	선택한 이름으로 저장합니다.

■ 컨트롤 박스 입력 포트 기능

컨트롤 박스의 입력 포트(Din 0 ~ Din 15)의 기능은 아래와 같은 특수 기능으로 설정 가능합니다(R = Rising Edge, F = Falling Edge, H = High State).

0. 기본 모드(GPIO)
1. 프로그램 시작 1 회 모드(Rising Edge)
2. 프로그램 중지(Rising Edge)
3. 프로그램 일시정지(Rising Edge)
4. R = 직접 교시 모드 켜기 / F = 직접 교시 모드 끄기
5. R = 속도 100% / F = 속도 0%
6. R = 리얼 모드로 전환 / F = 시뮬레이션 모드로 전환
7. R = 로봇 초기화
8. H = High State 인 경우 충돌 기능 끄기
9. H = 자동 초기화 키 (Auto-Initialization Key)
10. R = 외부 충돌로 인한 일시 정지 상태 해제
11. UI 프로그램에 Point 기능 추가 (Rising Edge)
12. 프로그램 시작 반복 모드 (Rising Edge)
13. R=Begin 위치로 이동 / F=이동 정지
14. R = 프로그램 재개
15. H = Quick Freedrive Change
16. R = 프로그램 일시 정지 / F = 프로그램 재개
17. F = 프로그램 일시 정지 / R = 프로그램 재개
18. H = 속도 조절 바 100% / L = 속도 조절 바 0%
19. R = 기본 설정 프로그램 로드
20. F = 로봇 팔 전원 끄기
21. R = Touch Sensing
22. F = Touch Sensing
23. H = No Arc
24. H = Program Start Block
25. R = Ext.Joint0 Plus/F=stop
26. R = Ext.Joint0 Minus/F=stop
27. R = Ext.Joint1 Plus/F=stop

28. R = Ext.Joint1 Minus/F=stop
29. R = Ext.Joint2 Plus/F=stop
30. R = Ext.Joint2 Minus/F=stop
31. H = Safety Speed
32. F = UserCoord0 ←TCP frame
33. F = UserCoord1 ←TCP frame
34. F = UserCoord2 ←TCP frame
35. F = Load & Run Program Table

주의

주의:

1) Digital 입력을 사용하기 전에 제조사가 제공하는 디지털 입력 포트의 전기적 특성을 충분히 숙지하고 사용하시기 바랍니다.

■ 컨트롤 박스 출력 포트 기능

컨트롤 박스의 출력 포트(Dout 0 ~ Dout 15)의 기능은 아래와 같은 특수 기능으로 설정 가능합니다(R = Rising Edge, F = Falling Edge, H = High State).

특수 기능으로 설정된 출력 포트는 티칭 프로그램에서 일반적인 용도로 사용하지 못합니다.

0. 기본 모드(GPIO)
1. H = 프로그램/로봇 실행 중 / L = Idle 상태
2. L = 프로그램/로봇 실행 중/ H = Idle 상태
3. H = 외부충격 감지
4. H = 직접 교시 모드 실행 상태
5. 선택 포트와 같은 번호의 디지털 입력 신호 전달
6. 툴 플랜지 입력 포트 0 번 신호 전달
7. 툴 플랜지 입력 포트 1 번 신호 전달
8. H = 로봇 활성화 상태 / L = 활성화되지 않음
9. H = 리얼 모드 상태 / L = 시뮬레이션 모드 상태
10. H = 로봇 동작 중 / L = Idle 상태
11. L = 로봇 동작 중 / H = Idle 상태
12. H = 로봇 초기화 (활성화) 실패 상태
13. H = 로봇 팔에 전원 인가 상태 / L = 팔에 전원 비인가 상태
14. H = 충돌 감지 켜져있는 상태 / L = 충돌 감지 기능 꺼져있는 상태
15. H = 일시 정지 상태
16. H = Inbox 0 번에 Trap 이 발생한 상태
17. H = Inbox 1 번에 Trap 이 발생한 상태
18. PWM 모듈로 사용
19. H = 티칭팬던트가 연결됨
20. H = 프로그램이 MAKE 창에서 실행됨
21. H = 프로그램이 PLAY 창에서 실행됨
22. H = 컨베이어 모드 실행 중
23. H = 컨트롤 박스 부팅 완료
24. H = 힘 제어 모드 실행 중

25. PC Alive Pulse
26. H = 속도 조절 바 100%
27. H = 최근 프로그램 불러오기 성공
28. H = TCP is in InBox 0
29. H = TCP is in InBox 1
30. H = Is Alarm
31. H = Robot posture is Begin posture
32. H = Emergency Teaching Enable
33. H = Prog. Run in Sub.P area

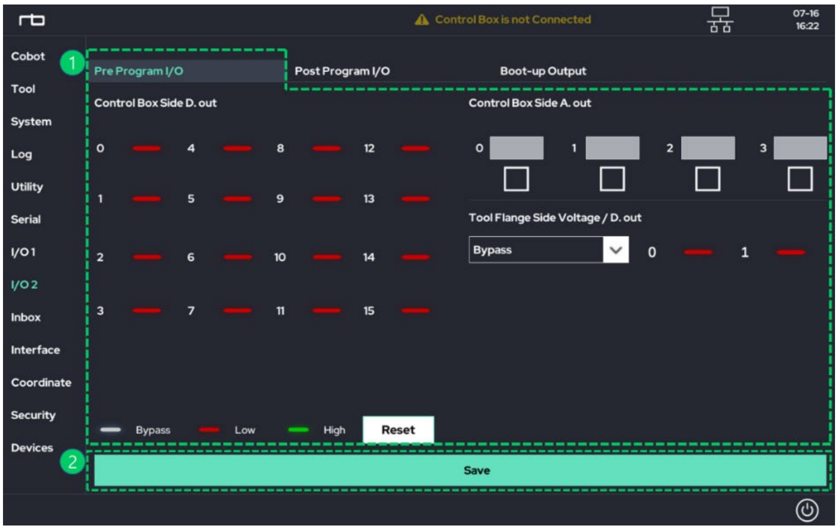
주의

주의:

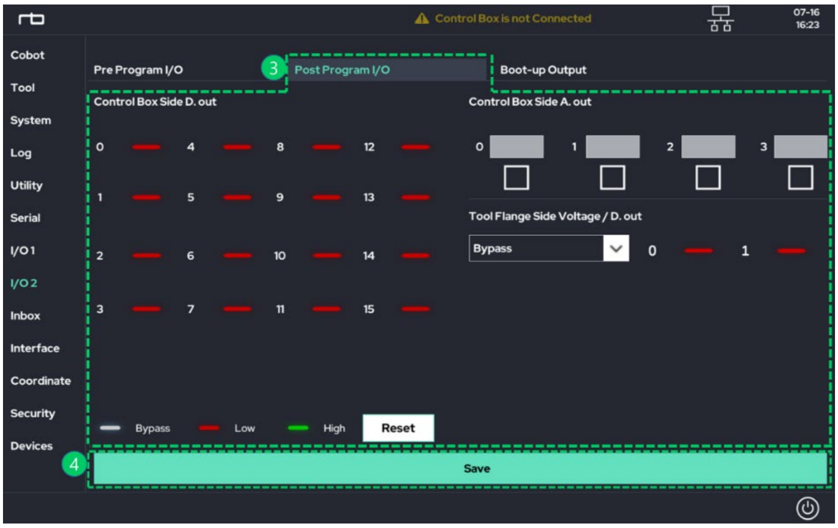
1) Digital 출력을 사용하기 전에 제조사가 제공하는 디지털 출력 포트의 전기적 특성을 충분히 숙지하고 사용하시기 바랍니다.

9.8 SET-UP(I/O 2)

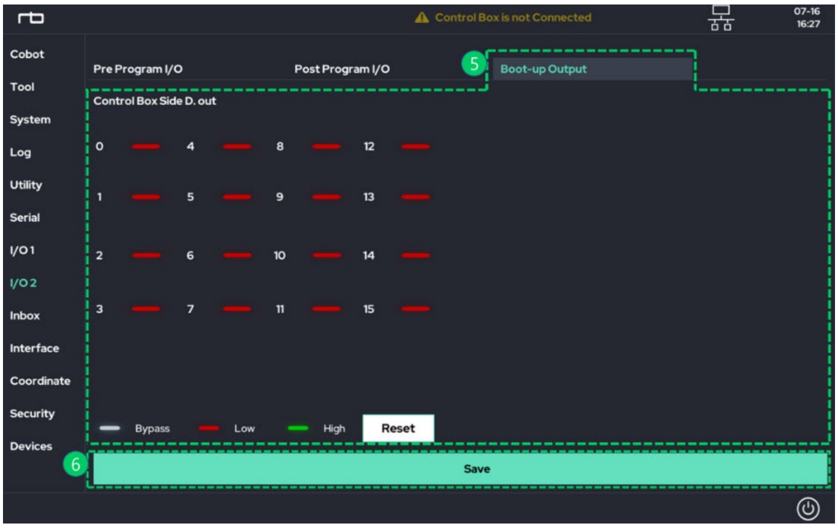
프로그램 시작 전 / 후에 항상 수행할 I/O 값을 설정합니다.



①	프로그램 시작 (재생)전에 수행할 I/O 송출을 설정합니다. 이곳에서 설정된 포트들은 프로그램 시작과 동시에 해당 설정 값으로 출력을 내보냅니다.
②	변경된 세팅을 저장합니다.



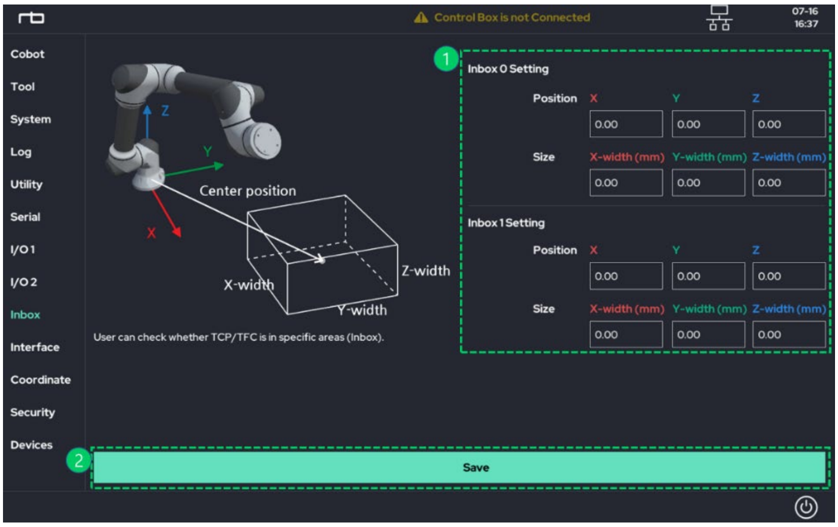
③	프로그램 종료 후에 수행할 I/O 송출을 설정합니다. 이곳에서 설정된 포트들은 프로그램 종료 동시에 해당 설정 값으로 출력을 내보냅니다.
④	변경된 세팅을 저장합니다.



⑤	제어 박스가 최초 부팅 직후, 제어 박스의 디지털 출력 옵션을 선택합니다.
⑥	변경된 세팅을 저장합니다.

9.9 SET-UP(INBOX)

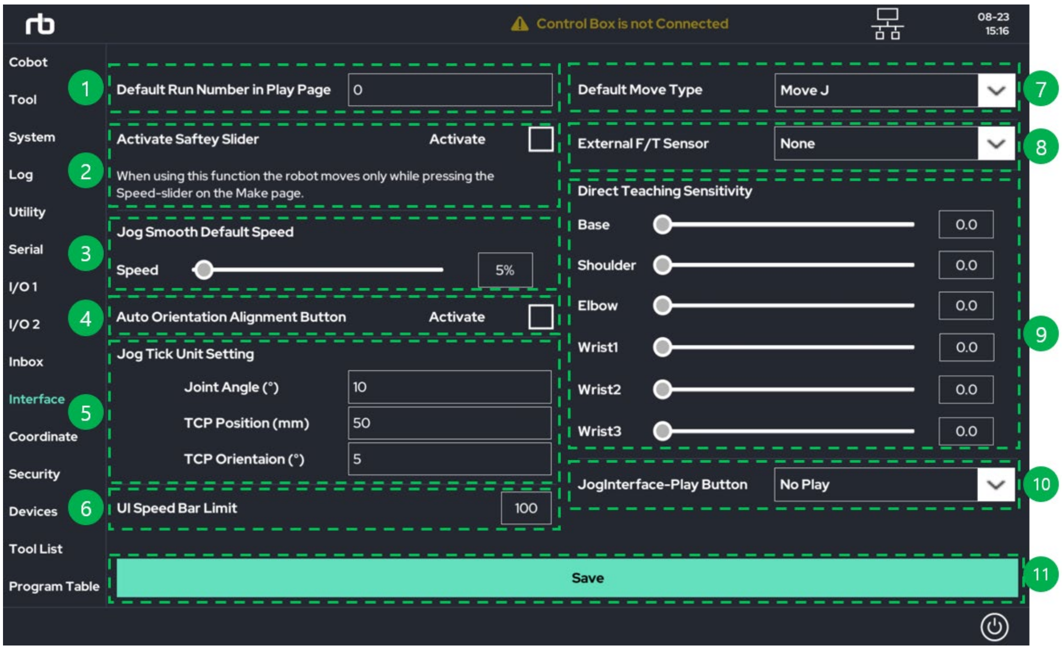
Inbox 기능 사용을 위해 Inbox 의 크기와 위치 정보를 세팅합니다.



①	Inbox 0 번과 1 번의 중심점 위치 및 사이즈를 입력합니다. 설정의 기준 좌표계는 제조사 베이스 좌표계입니다.
②	변경된 세팅을 저장합니다.

9.10 SET-UP(INTERFACE)

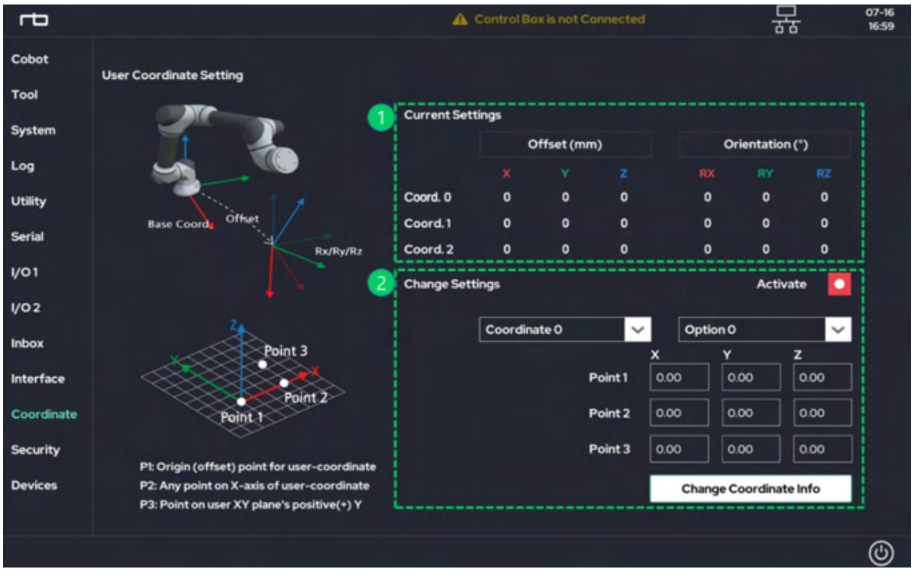
사용자가 로봇 및 UI 를 운용하는데 필요한 세팅을 수행합니다.



①	Play 화면에서 반복할 횟수를 입력합니다.
②	Make 화면에서 안전 슬라이더 기능을 사용할지 선택합니다. 안전 슬라이더 기능 사용 시, 사용자가 화면에서 손을 때면, 속도 조절 바가 0 으로 돌아옵니다. 안전 슬라이더 기능 해제 시, 사용자가 원하는 레벨로 속도 조절 바를 세팅해 놓고, 손을 화면에서 때면, 해당 속도 조절 바 옵션이 유지됩니다.
③	Make 화면 조그 Jog 기능의 Smooth 옵션 사용시 사용할 속도를 정합니다.
④	Make 화면 조그 하단에, 자동 회전 정렬 기능 버튼의 시각화 유무를 선택합니다.
⑤	Make 화면 조그 Jog 기능의 Tick 옵션 사용시 얼마만큼씩 이동할 지 세팅합니다.
⑥	UI 상의 속도 조절바 상한 값을 제한합니다.
⑦	Make 화면에서 Move 생성시 기본으로 생성할 동작 속성을 선택합니다.
⑧	외부 F/T 센서 사용 여부를 선택합니다.
⑨	Make 화면 조그 하단에, 자동 회전 정렬 기능 버튼의 시각화 유무를 선택합니다.
⑩	레인보우 로보틱스에서 제공하는 전용 조그/비상정지 인터페이스 장치를 사용할 경우, 장치에 있는 재생 버튼의 역할을 정의합니다.
⑪	변경된 세팅을 저장합니다.

9.11 SET-UP(COORDINATE)

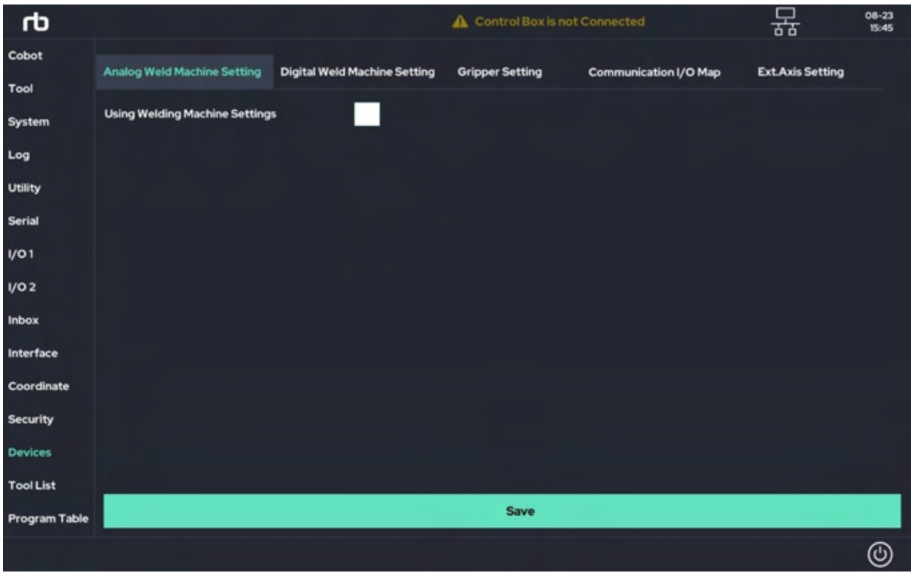
사용자 정의 좌표계 (User-Coordinate)를 설정합니다.



①	현재 세팅 된 사용자 정의 좌표계 정보입니다. 표시된 정보는 제조사 로봇 베이스 좌표계를 기준으로 합니다.
②	사용자 정의 좌표계의 설정을 변경합니다. 3 점 설정을 통해 좌표계를 설정합니다.

9.12 SET-UP(DEVICES)

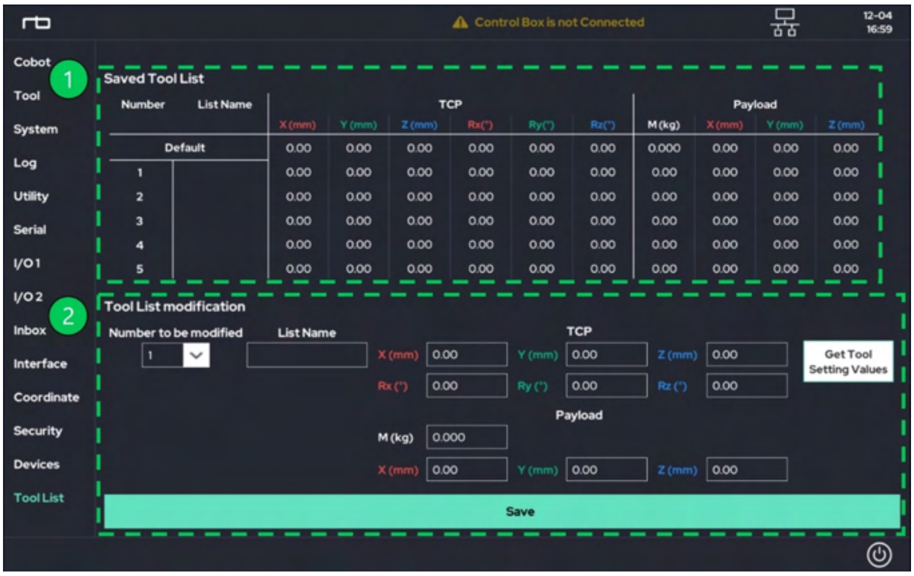
로봇과 연결된 추가 장비(Devices)를 설정합니다.



로봇과 연결된 장비에 대한 설정을 진행합니다.

9.13 SET-UP(TOOL LIST)

TCP 목록(Tool List)를 설정합니다.



①	현재 세팅 된 TCP 값을 나타냅니다.
②	수정하고자 하는 TCP 번호를 선택한다. 이름, TCP 위치 및 무게 중심 등을 설정한 뒤 저장한다.

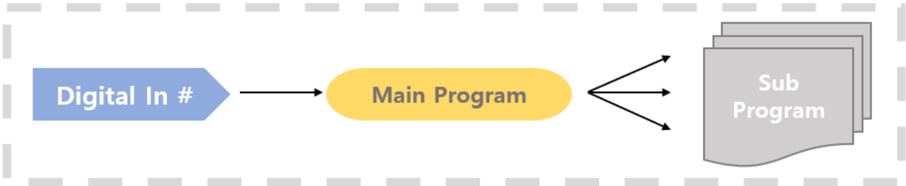
9.14 SET-UP(PROGRAM TABLE)

각각의 디지털 입력을 통해 불러올 프로그램을 설정합니다.

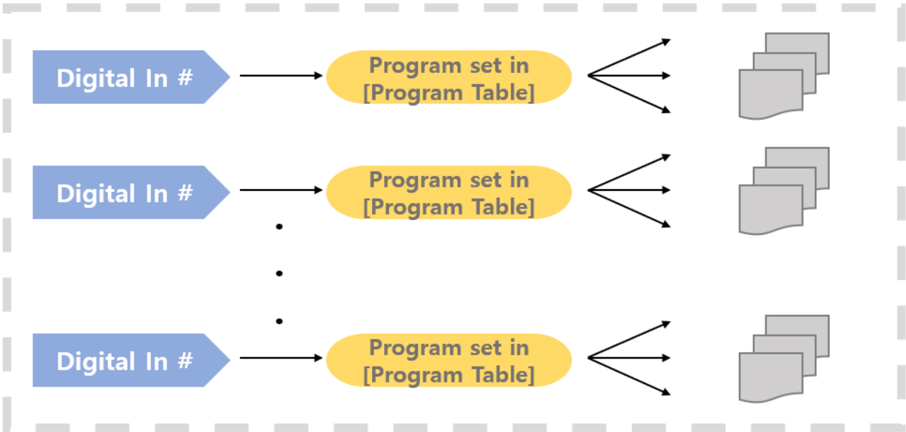
①	기존에 만들어 놓은 설정을 가져옵니다.
②	사용할 디지털 입력 포트를 의미합니다.
③	사용할 기능을 선택합니다. 기능은 Load, Load + Play(Once), Load + Play(Repeat)으로 나뉩니다.
④	디지털 입력을 통해 사용할 프로젝트를 선택합니다.
⑤	설정한 내용을 저장합니다.

기존의 'Setup > I/O1'의 '제어박스 디지털 입력 기능 정의'로 사용하는 Start Program 기능은 한 개의 디지털 입력만 사용 가능하며 프로그램도 제어 박스에 현재 업로드 되어 있는 메인 프로그램만 실행 할 수 있는 기능입니다. 'Setup > Program Table'은 서로 다른 디지털 입력에 서로 다른 프로그램을 로드, 프로그램 실행 등을 진행할 수 있습니다.

아래 그림을 통해 두 기능의 차이점을 확인할 수 있습니다.



SetUp > I/O1



SetUp > Program Table

제 10 장 유지 관리

10.1 점검 항목과 주기

장기간 로봇을 최상의 상태로 유지하기 위해서는 지속적인 점검이 필요합니다. 점검 담당자가 점검 계획을 작성하여 실시합니다. 다음과 같은 항목에 대한 주기적인 점검이 필요합니다. 점검 중 자체적으로 해결할 수 없는 문제를 발견했을 때는 제조사에 문의하십시오.

<table>
<tr><th>점검항목</th><th colspan="2">점검사항</th><th>주기</th></tr>
<tr><td rowspan="5">로봇 팔</td><td rowspan="3">전체</td><td>로봇에 적용한 프로그램에 따라 의도한 위치로 이동하는지 확인하십시오</td><td rowspan="2">매일</td></tr>
<tr><td>전원을 켜고 끌 때 로봇 팔이나 툴이 아래로 쳐지지 않는지 확인하십시오.</td></tr>
<tr><td>얼룩, 먼지 등 오염물질을 제거하여 주십시오.</td><td>3 개월</td></tr>
<tr><td>모터</td><td>이상 발열이나 소음이 발생하는지 확인하십시오.</td><td>매일</td></tr>
<tr><td>주요볼트</td><td>로봇 외부에 노출되는 볼트가 느슨해 졌는지 확인하십시오</td><td>3 개월</td></tr>
<tr><td rowspan="2">컨트롤 박스</td><td>케이블</td><td>케이블 연결 상태를 확인하십시오.</td><td rowspan="2">6 개월</td></tr>
<tr><td>내부</td><td>내부의 먼지를 제거하여 주십시오.</td></tr>
</table>

주의:

1) 유지 보수 시, 시스템(컨트롤 박스 및 로봇팔)에 연결된 전원을 차단하고 작업을 수행하세요.
2) 공압 / 전선 통과 모델의 경우 연결된 에너지원(공압 / 전원)을 제거하고 작업을 수행하세요.

10.2 로봇 팔 점검하기

■ **검사 주기**

최소 1 년마다 검사하며, 점검 부위마다 필요한 점검 주기는 다를 수 있습니다.

■ **로봇 팔 확인 및 청소하기**

1. 로봇 팔의 자세를 초기(Home) 위치로 전환하십시오.
2. 로봇 컨트롤 박스의 전원을 끄십시오.
3. 다음의 사항을 확인하십시오.
   ① 로봇 컨트롤 박스와 로봇 팔을 연결하는 케이블의 이상 여부
   ② 외부에 노출된 나사가 느슨해지거나 풀리지 않았는지 여부
   ③ 모터, 브레이크, 감속기 등의 이상 유무
4. 얼룩이나 먼지 등 오염 물질이 있는 경우 제거하십시오.

10.3 로봇 컨트롤 박스 점검하기

로봇 컨트롤 박스 내부에 먼지가 쌓일 경우 정전기 및 과열로 인해 제품이 오작동할 수 있습니다. 주기적으로 로봇 컨트롤 박스 내부를 확인하여 먼지를 청소하고 내부에 연결된 선들이 정상적으로 잘 연결되어 있는지 확인하십시오.

- **검사 주기**

  최소 6 개월마다 검사합니다. 검사 주기는 제품이 설치된 장소의 환경에 따라 달라 질 수 있습니다.

- **로봇 컨트롤 박스 확인 및 청소하기**

  1. 로봇 컨트롤 박스의 전원을 끄십시오.
  2. 로봇 컨트롤 박스의 덮개를 여십시오.
  3. 로봇 컨트롤 박스 내부에 먼지가 쌓여 있는지 확인하십시오.
  4. 먼지가 있을 경우 진공 청소기 등을 이용하여 조심스럽게 먼지를 제거하십시오.
  5. 커넥터의 선이 빠지지 않고 잘 연결되어 있는지 확인하십시오.

부록 A 시스템 기술 사양

로봇 팔

구분	사양
가반하중	RB5-850E Series: 5 kg / 11 lbs RB3-1200E Series: 3 kg / 6.6 lbs RB10-1300E Series: 10 kg / 22 lbs
중량	RB5-850E Series, RB3-1200E Series: 22 kg / 48.5 lbs RB10-1300E Series: 33 kg / 72.8 lbs
도달 범위	RB5-850E Series: 850 mm / 33.5 in RB3-1200E Series: 1200 mm / 47.2 in RB10-1300E Series: 1300 mm / 51.1 in
자유도	6 축
조인트 범위	± 360° (Elbow: ± 165°)
속도	조인트: 최대 180° /s, 툴: 1m/s
반복 정밀도	±0.1 mm
설치면적	RB5-850E/RB3-1200E Series: Φ173 mm RB10-1300E Series: Φ196 mm
툴(엔드 이펙터) 커넥터 타입	M10 12-pin
툴 I/O 포트	Non-E 버전: 디지털 입력 2, 디지털 출력 2, 아날로그 입력 2 E 버전: 디지털 입력 6, 디지털 출력 2
툴 통신	RS485
툴 전원	12V/24V, 2A
IP 등급	IP66
사용온도 / 소음	0 ~ 50 °C / <65dB
재질	알루미늄, 스틸.
케이블 길이	전원 케이블, 로봇 팔 – 컨트롤 박스 연결 케이블, 비상정지 /조그 인터페이스 케이블 : 5m

스탠드 타입 컨트롤 박스

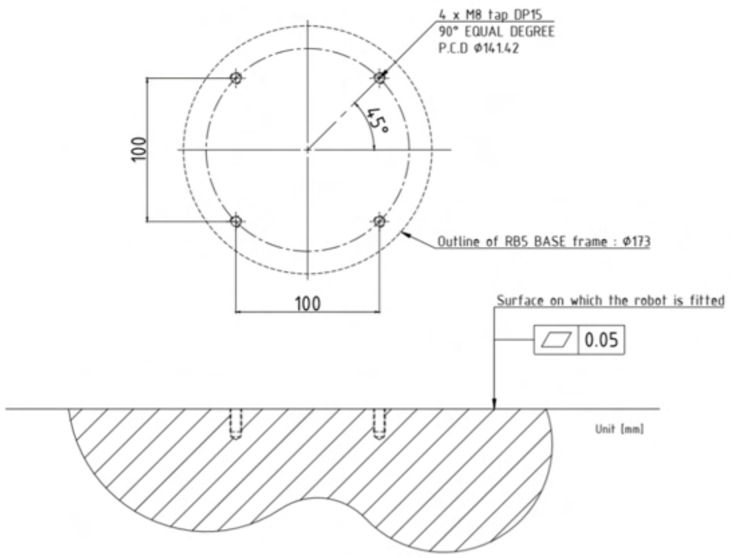
구분	사양
중량	CB04: 17.3 kg / 38.14 lbs
크기(W x H x D)	454 x 240 x 416.2 mm
I/O 포트	디지털 입력 16, 디지털 출력 16 아날로그 입력 4, 아날로그 출력 4
통신	Ethernet, TCP/IP
전원	100 ~ 240 VAC, 50 ~ 60 Hz
재질	EGI

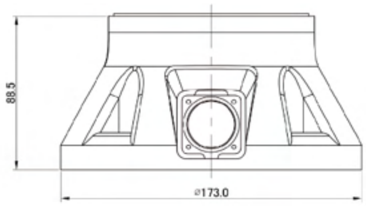
부록 B 로봇 설치면 치수

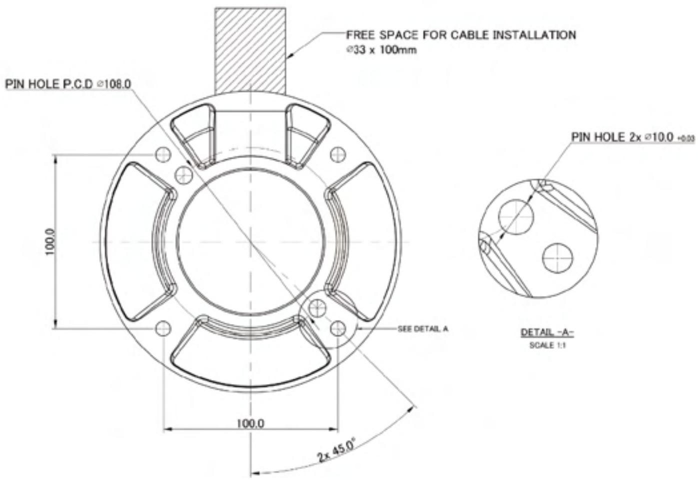
■ RB5-850E / RB3-1200E Series 로봇 설치면 치수

▶ P.C.D: Pitch Circle Diameter

▶ DP: Depth



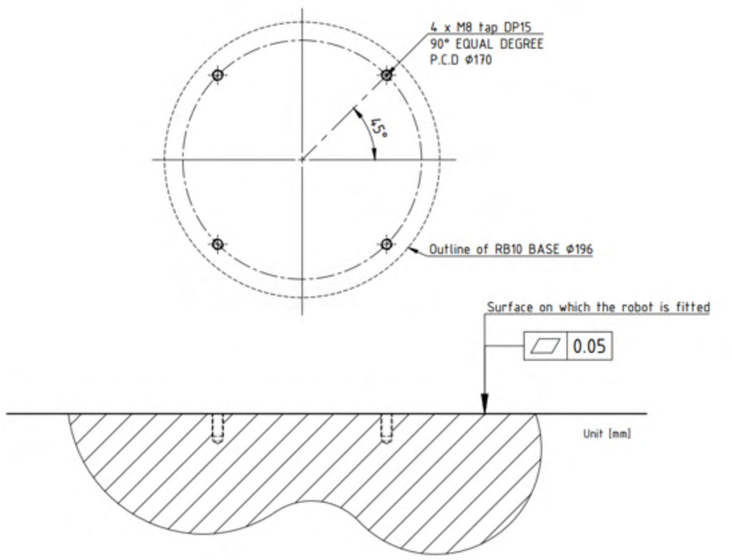


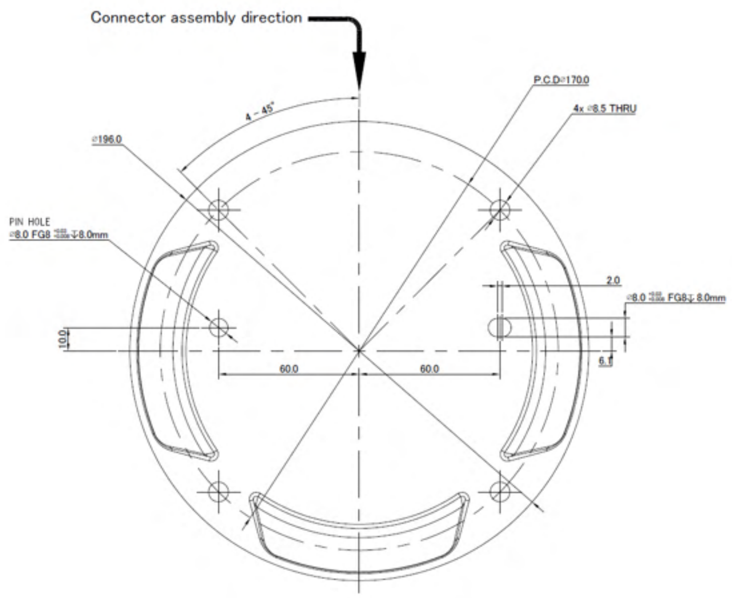


■ RB10-1300E Series 로봇 설치면 치수

▶ P.C.D: Pitch Circle Diameter

▶ DP: Depth.



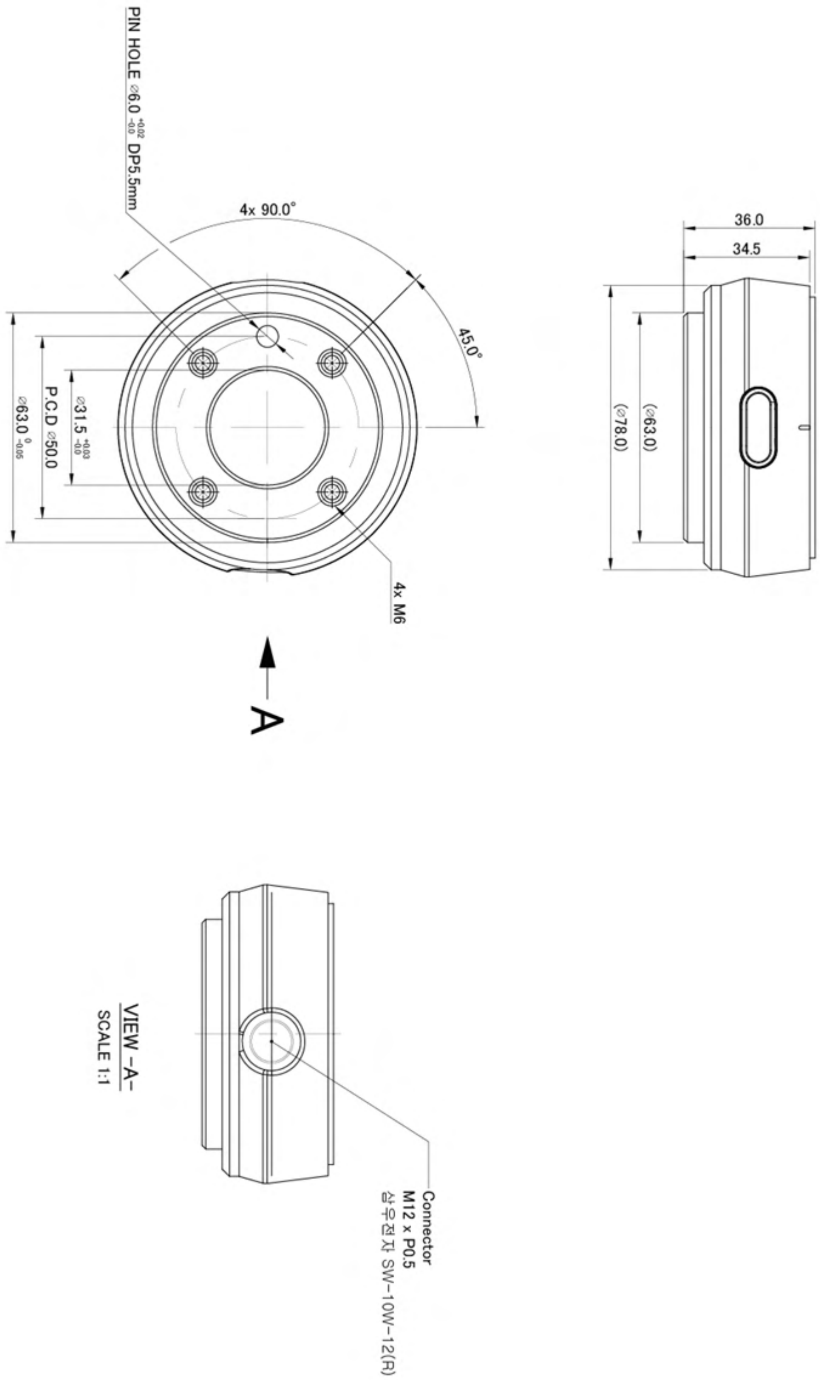


부록 C 툴 플랜지 단면도

- RB5-850E / RB3-1200E Series 툴 플랜지 단면도

▶ P.C.D: Pitch Circle Diameter

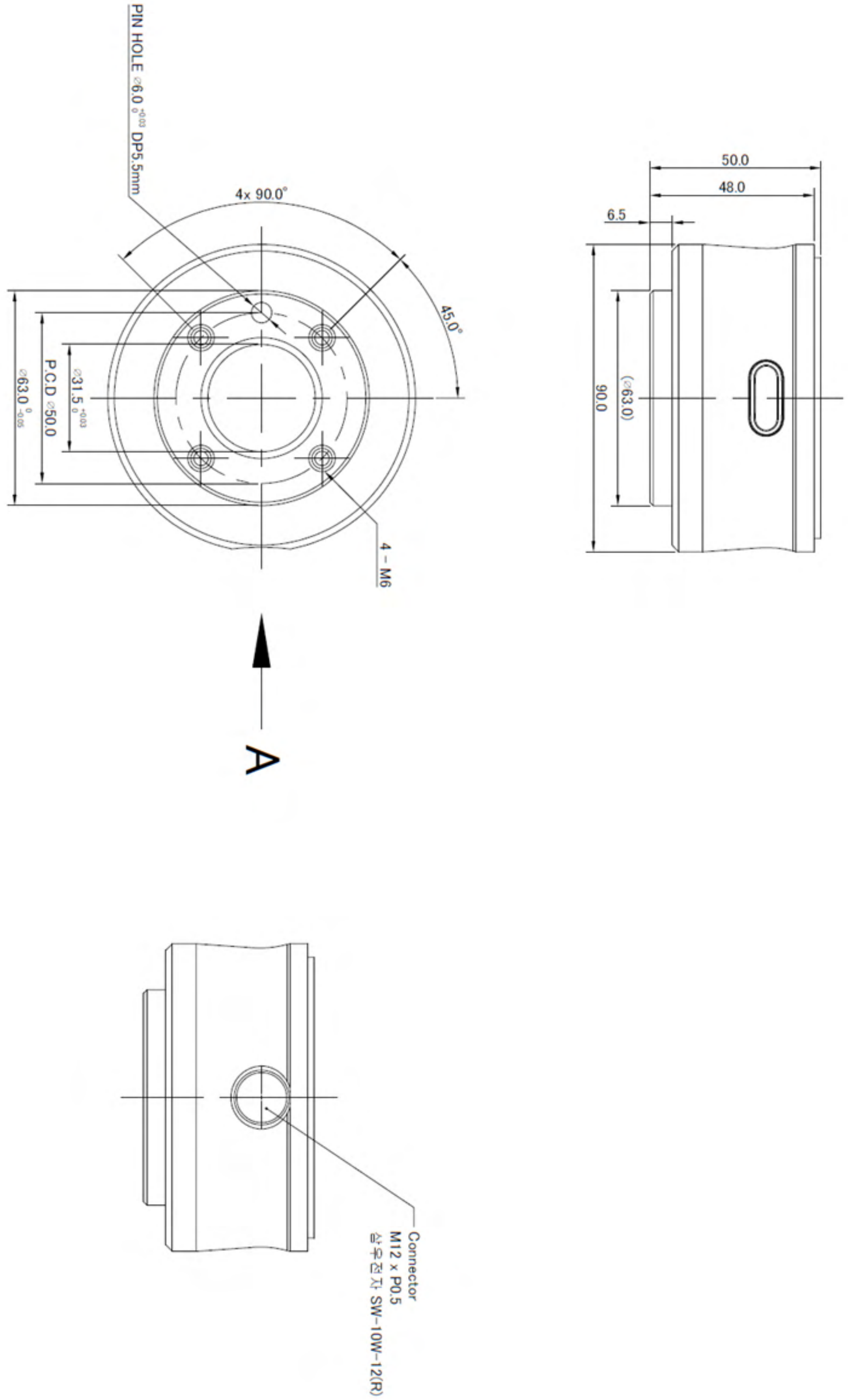
▶ DP: Depth.



■ RB10-1300E Series 툴 플랜지 단면도

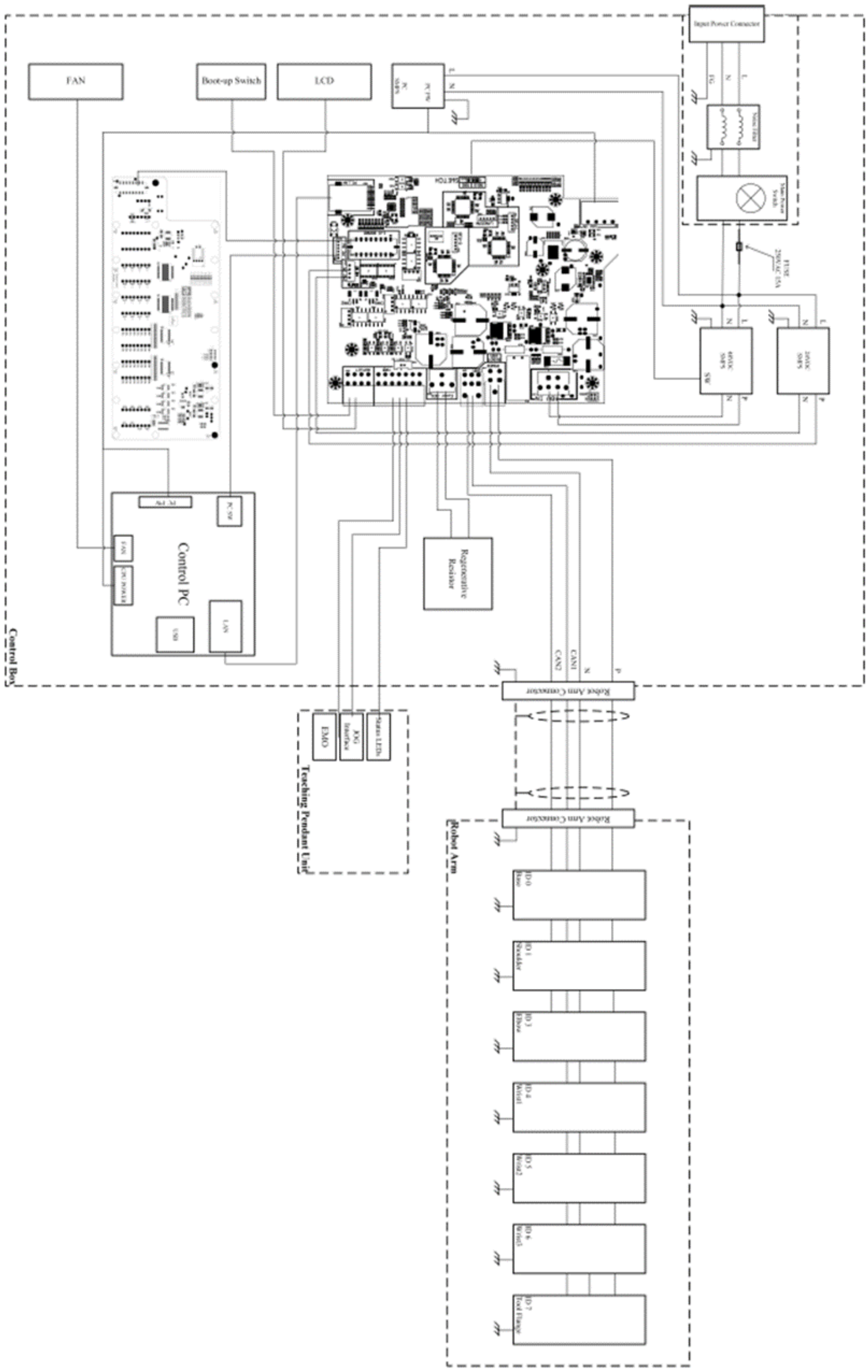
▶ P.C.D: Pitch Circle Diameter

▶ DP: Depth.



부록 D 컨트롤 박스 전기 도면

■ 스탠드 타입 컨트롤 박스 전기 도면

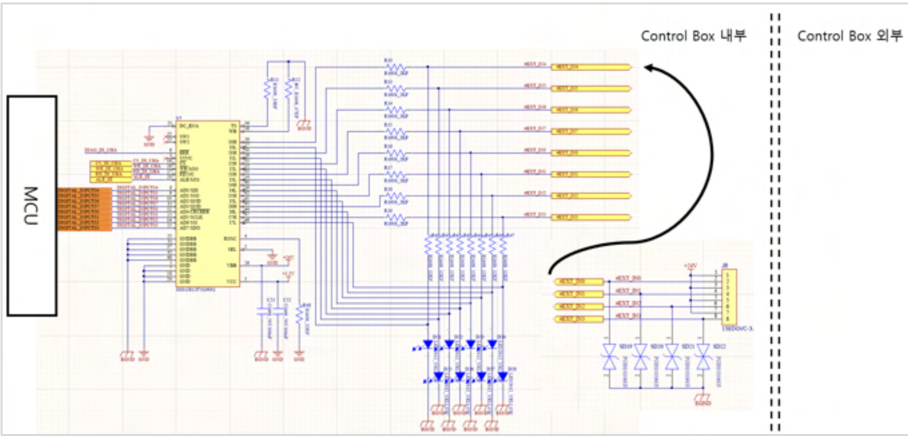


부록 D-1 컨트롤 박스 디지털 입력

- **주의**

  Control Box Digital 입력 포트 결선 이전에, 전원 차단이 되어야 합니다.

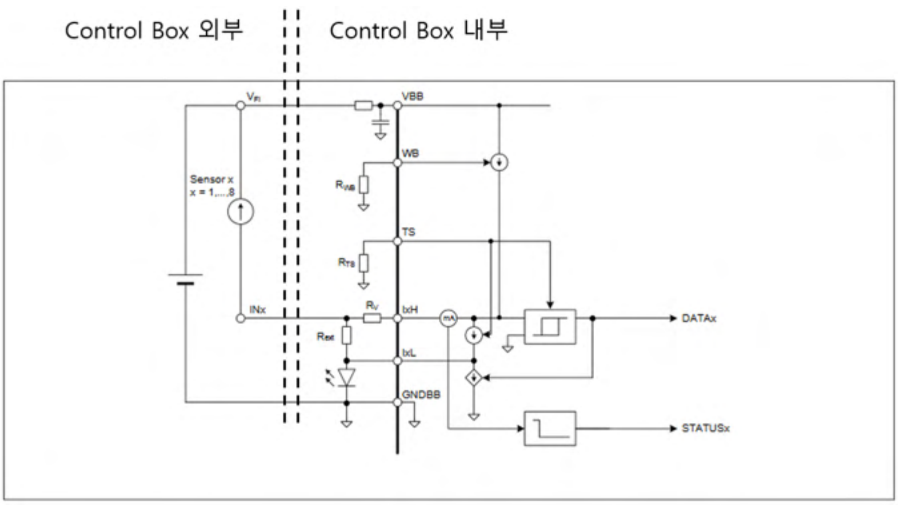
1. 디지털 입력[DI00 ~ DI15] 내부 회로도



Control Box Digital 입력[DI00-DI15]을 받는 소자 구성입니다.

내부 24V 공급 단자가 있으며, 외부 24V 공급 시 오동작이 발생합니다.

2. Digital 입력[DI00 ~ DI15] 소자 사용 방식



RB Control Box Digital 입력 소자[DI00-DI15] 사용 방법입니다.

Functional Description

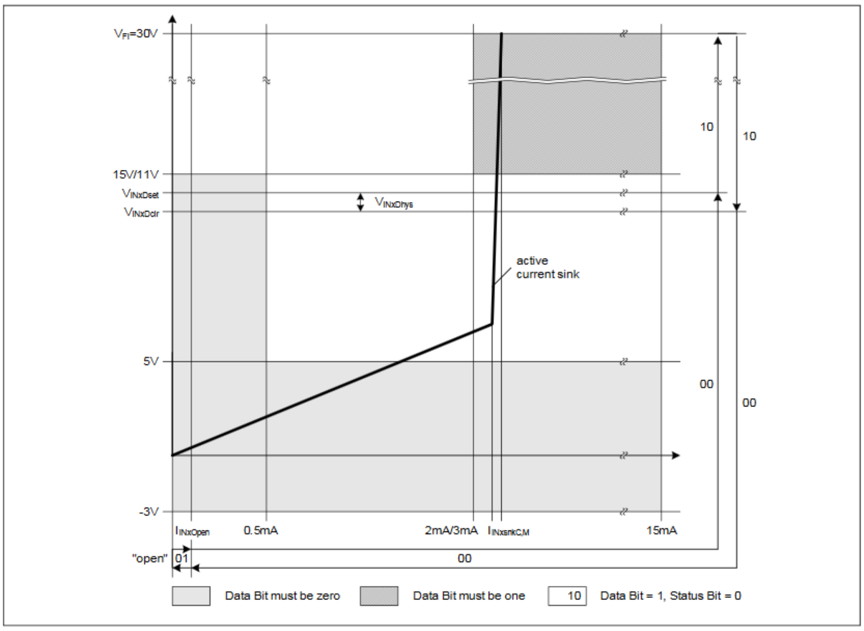


Figure 11 Sensor Input Characteristics

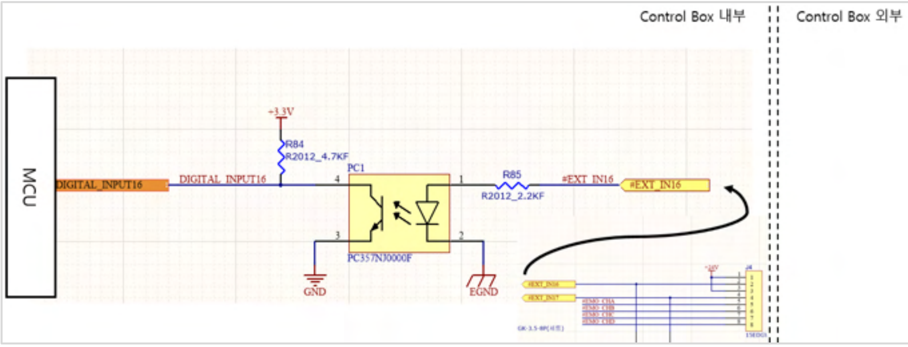
Digital 입력 신호의 전압/전류 특성 곡선입니다.

3. Digital 입력[DI00 ~ DI15] 특성

Terminals	Parameter	Min	Typ	Max	Unit
[DI00 – DI15]	Voltage	-3	-	30	V
[DI00 – DI15]	OFF region	-3	-	5	V
[DI00 – DI15]	ON region	11	-	30	V
[DI00 – DI15]	Current(11-30V)	2	-	15	mA
[DI00 – DI15]	Function	-	PNP+	-	Type
[DI00 – DI15]	IEC 61131-2	-	1	-	Type

이는 Digital 입력 0 번부터 Digital 입력 15 번까지만 적용되는 사양입니다.

4. Digital 입력[DI16-DI17] 내부 회로도



Control Box Digital 입력(DI16-DI17)을 받는 소자 구성입니다.

내부 24V 공급 단자가 있으며, 외부 24V 공급 시 오동작이 발생합니다.

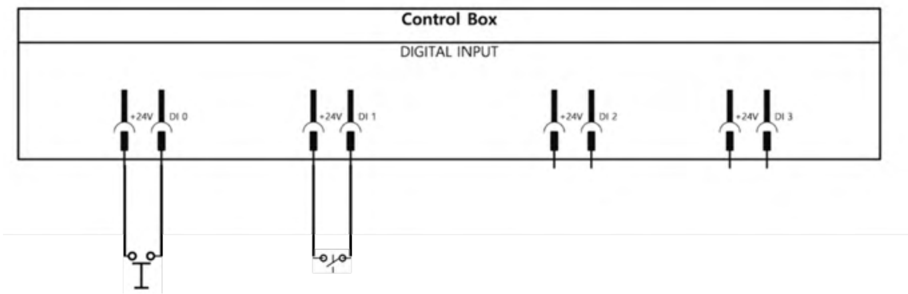
5. Digital 입력[DI16-DI17] 특성

Terminals	Parameter	Min	Typ	Max	Unit
[DI16 – DI17]	Voltage	0	-	25	V
[DI16 – DI17]	OFF region	0	-	7	V
[DI16 – DI17]	ON region	7	-	25	V
[DI16 – DI17]	Function	-	PNP+	-	Type

이는 Digital 입력 16, 17 번만 적용되는 사양입니다.

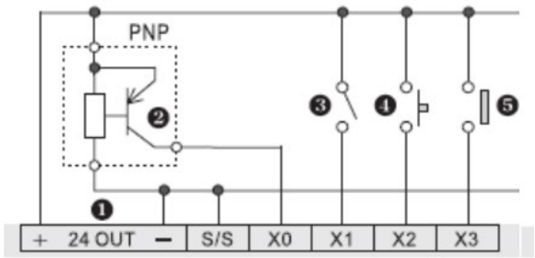
6. 테스트 환경

Digital 입력 소자 테스트는 Toggle 스위치를 사용하여 진행하였으며, 아래와 같은 구성으로 테스트하였습니다.



7. PNP 방식 센서 사용 방법

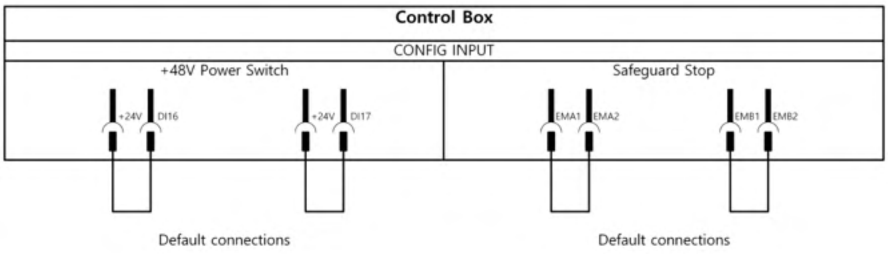
출처: https://blog.naver.com/mjg5080/97380010



PNP 방식 센서의 경우 위 결선과 동일하게 사용 가능합니다.

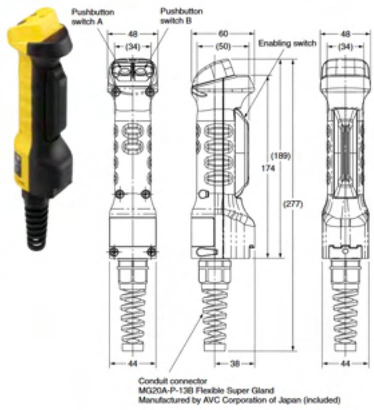
이는 모든 Digital 입력에 적용되는 사양입니다.

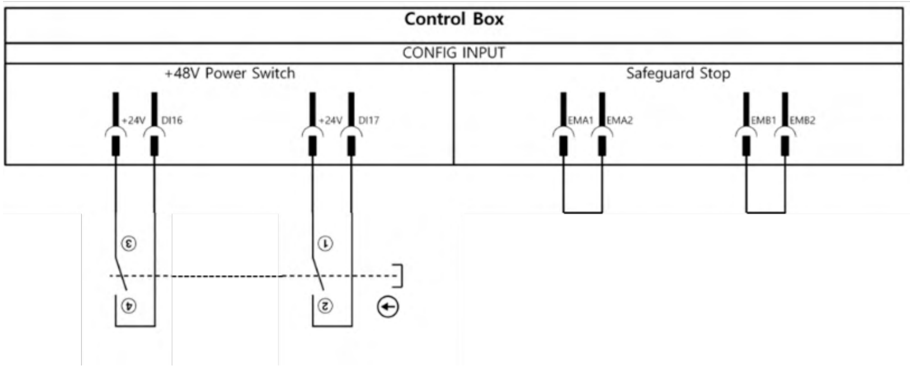
8. 3-Position Enabling Device 결선 방법

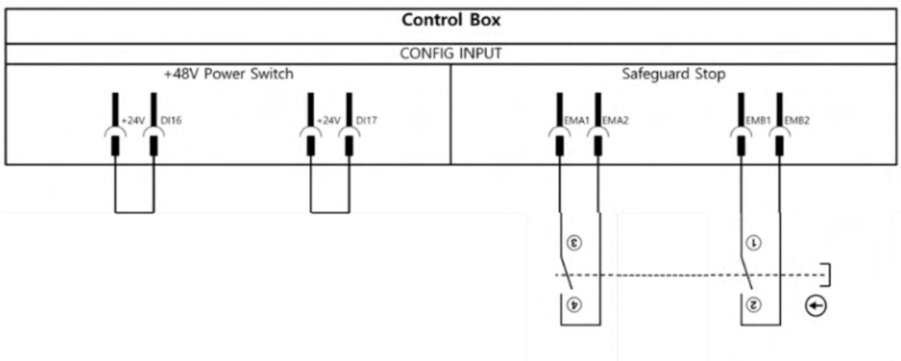


초기 출하 상태는 위와 같으며, 안전에 의거하여 동작허가 장치 장착이 가능합니다.

출처: https://www.motionsolutions.com







이는 ISO 10218, IEC 60204-1 에 의거한 동작허가 장치(Enabling Device)에 적용되는 내용입니다.

9. 안전 기기 결선 방법

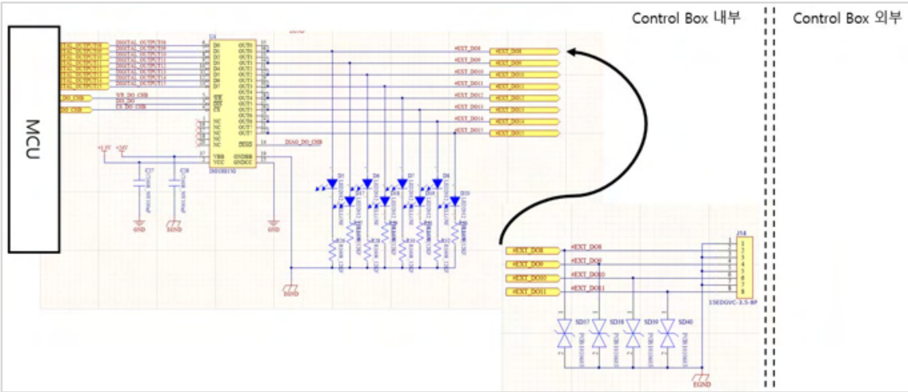
라이트 커튼, 안전 도어 센서 등과 같이 PNP 방식 센서 및 Enabling Device 등 같은 결선을 사용하는 안전 기기 결선은 위 설명과 동일 합니다.

부록 D-2 컨트롤 박스 디지털 출력

- **주의**

  Control Box Digital 출력 포트 결선 이전에, 전원 차단이 되어야 합니다.

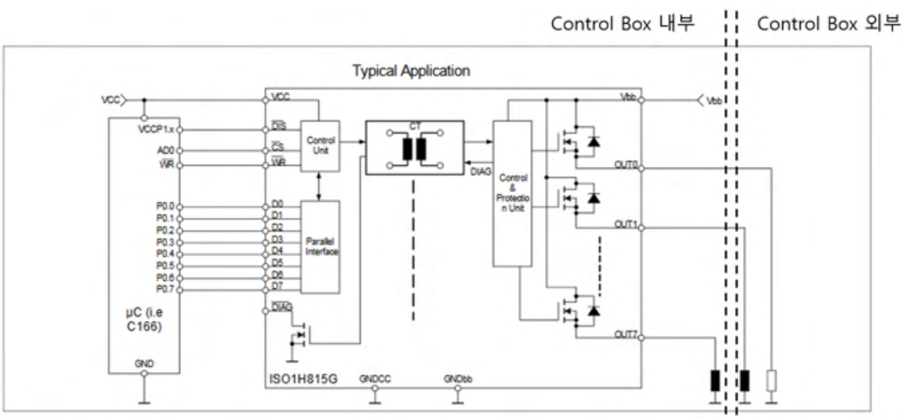
1. Digital 출력[DO00-DO15] 내부 회로도



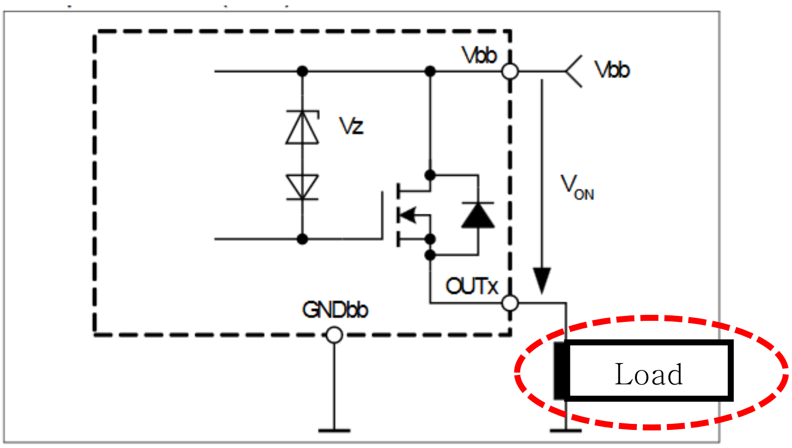
Control Box Digital 출력[DO00-DO15]을 하는 소자 구성입니다.

내부 GND 단자가 있으며, 연결하고자 하는 외부 센서, 장비의 GND 와 연결이 되어야 합니다.

2. Digital 출력[DO00-DO15] 소자 사용 방식



RB Control Box Digital 출력 소자[DO00-DO15] 사용 방법입니다.



단일 Digital 출력의 사용 방법입니다.

Control Box 내부에 Vbb 전원이 공급되고 있으며 소스 방식으로 출력 됩니다.

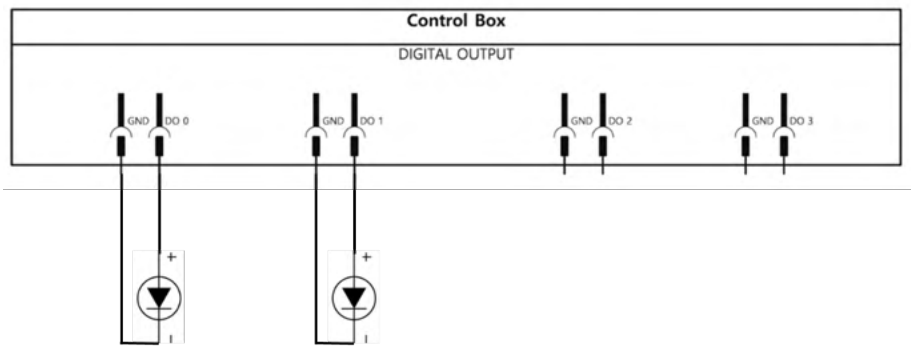
3. Digital 출력[DO00-DO15] 특성

Terminals	Parameter	Min	Typ	Max	Unit
[DO00 – DO15]	Voltage	-	24	-	V
[DO00 – DO15]	CURRENT	0	-	1	A
[DO00 – DO15]	Function	-	PNP	-	Type

단일 채널 1A 사용 가능하지만, 모든 채널의 총 전류량이 2A 이하가 되어야 한다.

4. 테스트 환경

Digital 출력 소자 테스트는 24Vdc LED 를 사용하여 진행하였으며, 아래와 같은 구성으로 테스트하였습니다.



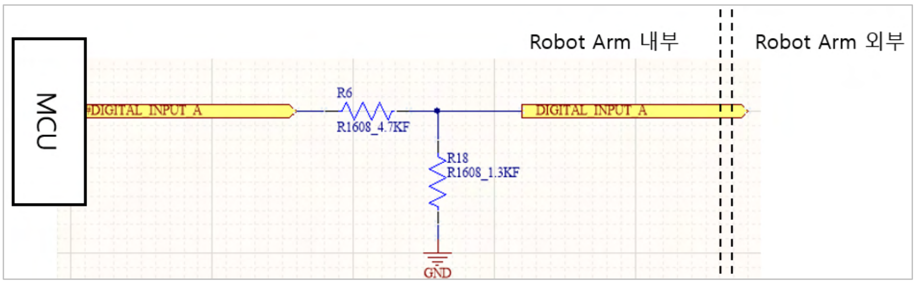
부록 D-3 툴 플랜지 디지털 입력

■ **주의**

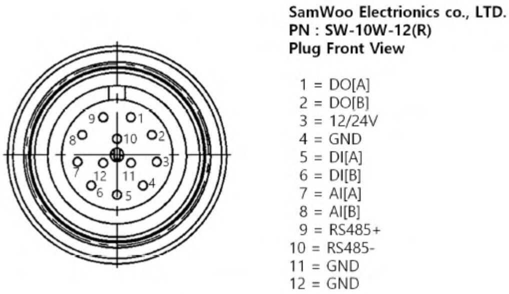
RB Series Tool Flange I/O 입력 포트 결선 이전에, 전원 차단이 되어야 합니다.

■ **하단의 전기 도면은 Non-E 타입에만 해당합니다.**

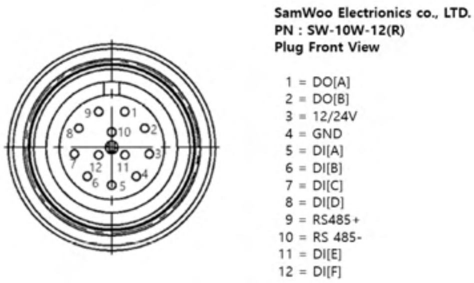
1. Digital 입력 내부 회로도



Tool Flange Digital 입력을 하는 소자 구성입니다.



(1) Non-E Version Robot



(2) E Version Robot

외부 노출된 커넥터 결선도입니다. 로봇 버전에 따라 결선도가 위와 같이 나뉘게 됩니다.

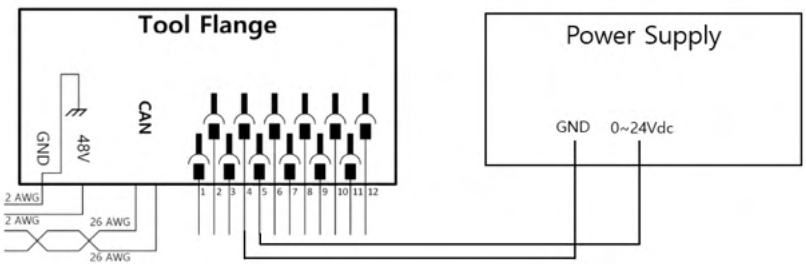
2. Digital 입력 특성

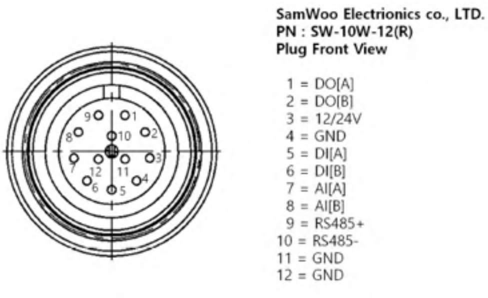
Terminals	Parameter	Min	Typ	Max	Unit
[DIA, ... , DIF]	Voltage	0	-	24	V
[DIA, ... , DIF]	OFF region	0	-	9	V
[DIA, ... , DIF]	ON region	10	-	24	V

이는 Tool Flange Digital 입력에만 적용되는 사양입니다(이때, Non-E version Robot 은 DIA, DIB 만 적용됩니다.)

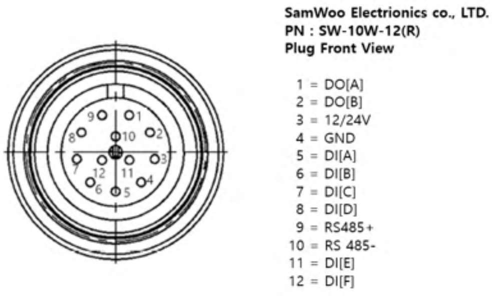
3. 테스트 환경

Digital 입력 소자 테스트는 파워 서플라이를 사용하여 진행하였으며, 아래와 같은 구성으로 테스트하였습니다.





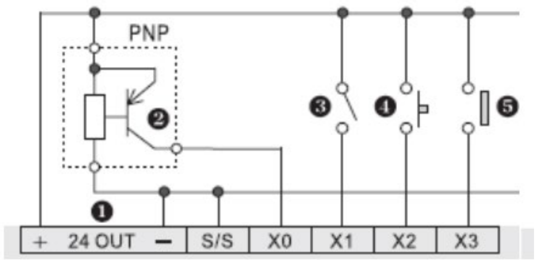
Non-E Version



E Version

4. PNP 방식 센서 사용 방법

출처: https://blog.naver.com/mjg5080/97380010



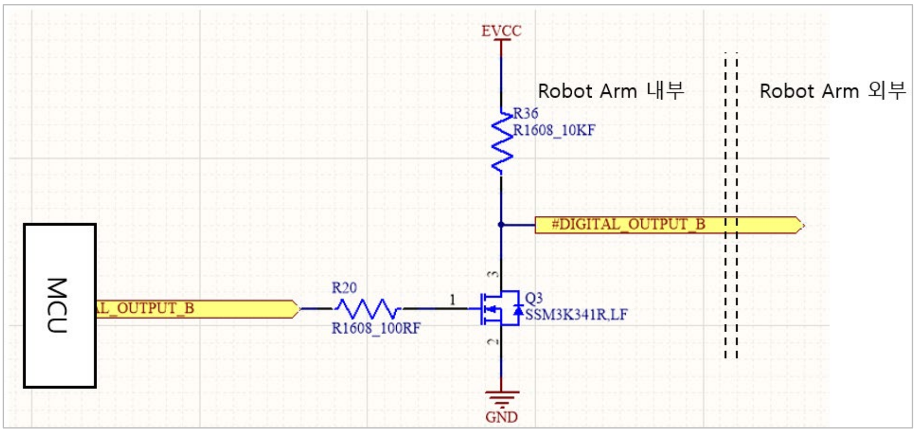
PNP 방식 센서의 경우 위 결선과 동일하게 사용 가능합니다.

이는 Control Box Digital 입력에 동일하게 적용되는 사양입니다.

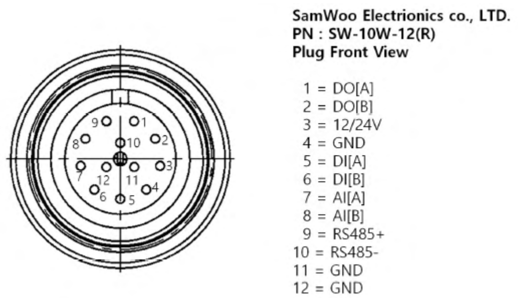
부록 D-4 툴 플랜지 디지털 출력

- **주의**
  Tool Flange I/O 출력 포트 결선 이전에, 전원 차단이 되어야 합니다.
- **하단의 전기 도면은 Non-E 타입에만 해당합니다.**

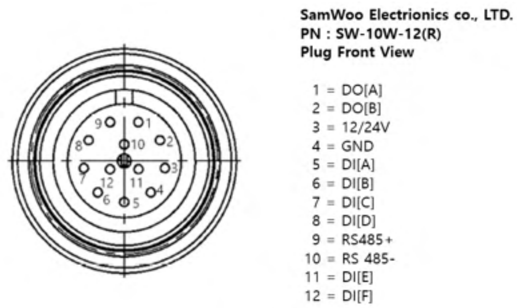
1. Digital 출력 내부 회로도



Tool Flange Digital 출력[DOA, DOB]을 하는 소자 구성입니다.



(1) Non-E Version Robot



(2) E Version Robot

외부 노출된 커넥터 결선도입니다. 로봇 버전에 따라 결선도가 위와 같이 나뉘게 됩니다.

2. Digital 출력 특성

Terminals	Parameter	Min	Typ	Max	Unit
[DOA, DOB]	Voltage	0	12/24	24	V
[DOA, DOB]	CURRENT Ver 1.	0	150	700*	mA
[DOA, DOB]	CURRENT Ver 2.	0	2000	2000	mA

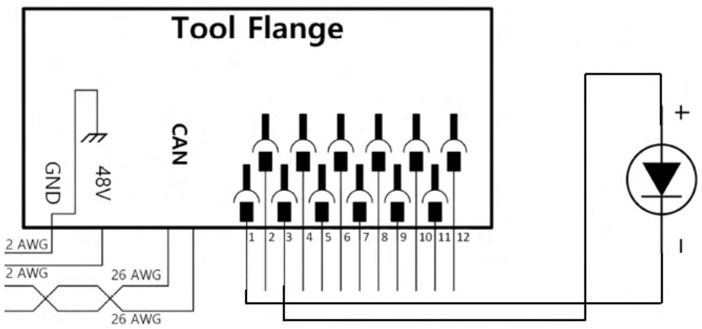
*Tsp=25°C; pulsed; tp≤10μs

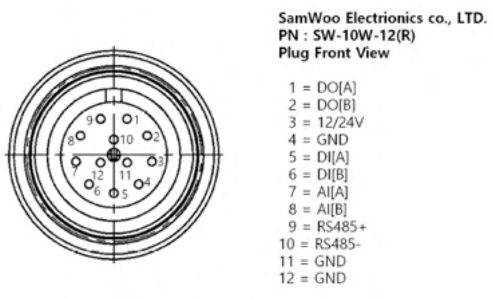
<u>이는 Tool Flange Digital 출력 A, B 만 적용되는 사양입니다.</u>

<u>2019. 07. 24. 일 기준 납품된 협동로봇은 Current Ver 1. 입니다.</u>

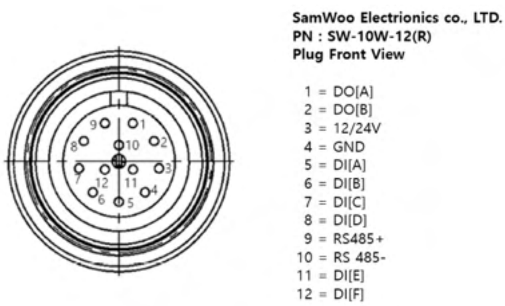
3. 테스트 환경

Digital 출력 소자 테스트는 24V dc LED 를 사용하여 진행하였으며, 아래와 같은 구성으로 테스트하였습니다.





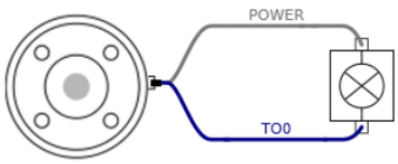
(1) Non-E Version



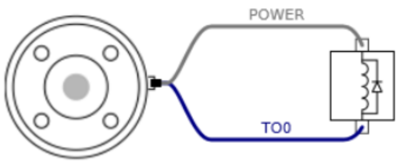
(2) E Version

자사 매뉴얼에 아래와 같은 예시가 정의되어 있습니다.

- 툴 디지털 출력을 사용하기 위해서는 아래 그림과 같이 12V 또는 24V 전원 공급을 사용하여 부하를 켜는 것을 보여줍니다. Tool Out 블록에서 출력 전압을 정의할 수 있습니다.



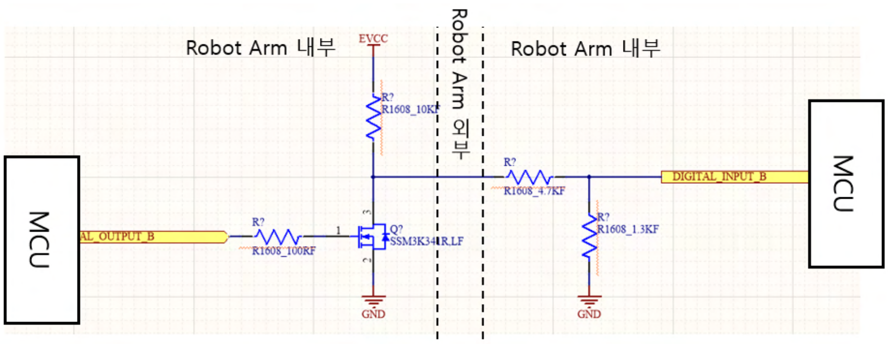
※ 아래 그림에 표시된 대로 유도 부하에 대하여 보호 다이오드를 사용하시길 적극 권장합니다.



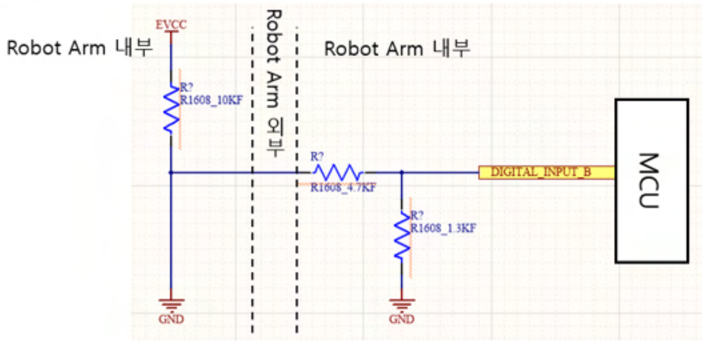
4. 사용시 주의 사항

Digital 출력 소자의 경우 NPN 방식이지만 내부 10K 풀업 저항이 내장되어 있습니다.

대부분 소자(LED, 솔레노이드 밸브, 릴레이)는 3 번 테스트 환경 혹은, 상용 Gripper 에 디지털 신호 인가 기능과 같이 사용이 가능하지만, 자사 Tool Flange Digital 입력 소자와 같은 전압 분배를 사용하는 환경에서는 동작이 안될 수 있습니다.

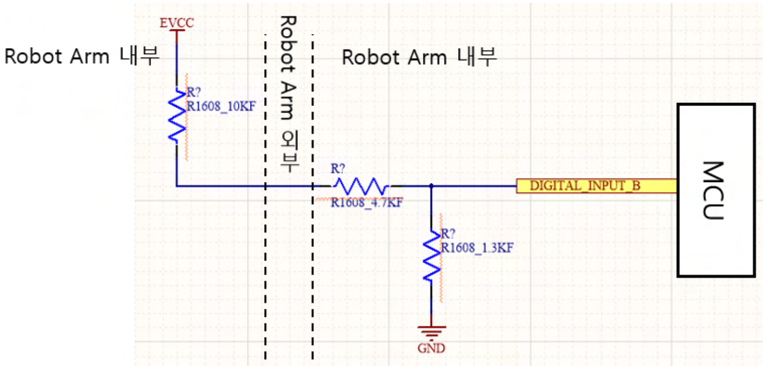


<u>자사 Digital 출력을 Digital 입력에 연결한 경우</u>



Digital 출력 Low

Digital 입력은 0V 가 입력되어 Low 로 출력 됩니다.



Digital 출력 High

Digital 입력은 저항 값에 따라 인식이 안될 경우가 생깁니다.

위 경우의 경우 MCU 에 인가되는 전압이 EVCC 24Vdc 일 경우 약 2V 가 인가되며, 이는 Low 로 인식이 됩니다.

부득이 하게 위 상황으로 운용해야 할 경우 Digital 입력 단 저항 비 조절이 필요 합니다.

부록 E 외부 스크립트 제어 API

E.0 개념

협동로봇 RB Series 는 다양한 환경과 목적으로 운용될 수 있습니다. 다수의 RB Series 혹은 다른 시스템과 연동되어 사용될 수 있습니다. 비전시스템과 연동되어 이동 좌표를 실시간으로 변경하거나 사용자가 기존에 사용하던 시스템의 한 부분으로 사용할 수도 있습니다.

주어진 태블릿 UI 로 로봇을 제어할 수도 있지만, 사용자의 편의나 운용의 이점을 위해서 외부의 임의의 제어기로부터 로봇을 제어할 수 있는 방법을 제공합니다.

RB Series 는 기본적으로 스크립트 명령을 받고 그 명령을 수행합니다. 태블릿 UI 로 태스크 모션을 작성하고, 그 파일의 스크립트를 차례대로 수행하는 것이 일반적인 운용방안이라면, 본 문서에서 설명하고자 하는 방법은 외부의 다른 디바이스로부터 명령어 스크립트를 입력 받아 해석하여 움직이는 방법입니다.

태블릿 UI 에서 제공하던 IF, REPEAT 등의 제어 구문은 사용자가 외부 제어 디바이스에서 직접 구현하고, 로봇의 동작 명령어와 IO 제어 명령어를 상황에 맞게 보내주어 로봇을 구동합니다.

본 문서에서는 위 개념으로 로봇을 구동하는 예제를 설명합니다.

E.1 외부 제어 스크립트 API

본 문서에서 제공하는 스크립트의 설명은 외부 제어를 위한 전용 스크립트로 태블릿을 사용해 작성하는 ".wsl" 워크 문서의 스크립트와 유사해 보이지만 다릅니다. 워크 문서에는 "repeat", "if-else", "break" 등과 같은 흐름을 제어하는 구문이 들어가기 때문에 하나의 문장(statement)의 완성이 동작으로 직결되지 않고, 그 문장의 상위 문장도 완성되어야 합니다.

예를 들어 move 명령 안에 point 들이 존재한다고 하였을 때 다음과 같이 표현될 수 있습니다.

```
1)
move joint {
   point ( ) absolute 0.4, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
      point ( ) absolute 0.4, 0.1, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0
}
```

```
2)
move joint {
   point ( ) absolute 0.4, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
      point ( ) absolute 0.4, 0.1, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0
```

1)과 2)의 차이는 마지막에 "}"의 유무입니다. 두 경우에서 point 문장은 완성이 되어 있습니다. 하지만 1)과 달리 2)는 point 의 상위 문장인 move 문장이 완성되지 않았기 때문에 동작할 수 없는 구문이고, 파서(parser)는 문장이 완성되길 기다릴 것입니다.

```
3)
folder( ) {
      move joint {
      point ( ) absolute 0.4, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
            point ( ) absolute 0.4, 0.1, 5.0, 5.0, 5.0, 5.0, 5.0, 5.0
      }
```

같은 논리로 3)의 경우에도 파서는 folder 문장이 완성되길 기다리며 동작되지 않습니다.

위와 같이 여러 줄의 구문들이 완성되어야 동작하는 방법은 외부 제어 방법에는 적합하지 않습니다. 사용자는 외부 제어를 통해 명령을 보내는 순간 그 명령을 파싱(parsing)해서 동작하길 기대합니다. 여러 번의 명령을 보내서 그 명령들이 3)과 같이 여러 줄의 문장들을 완성시켜 나가는 것을 원하지 않습니다.

그래서 외부 제어는 모든 명령이 한 문장으로 끝날 수 있도록 간결하게 구성되어 있습니다. 대신에 흐름을 제어하는 구문이나 기타 불필요한 기능들을 제공하지 않습니다. 워크 문서에 있던 "repeat", "if-else", "break"와 같은 구문이나 "wait"와 같은 구문들도 외부 제어를 수행하는 소프트웨어에서 자체적으로 해결하는 것이 구조상이나 논리상 더 적합하고 간결합니다.

우선 설명드릴 스크립트는 실제로 로봇을 움직이는 동작 명령어입니다.

동작 명령은 크게 5 가지입니다.

1) jointall
2) movetcp
3) movecircle
4) blend_jnt
5) blend_tcp

jointall

명령어	jointall
스크립트	jointall spd, acc, joint1, joint2, joint3, joint4, joint5, joint6
설명	이 명령은 로봇의 각 관절을 Joint Space 에서 움직입니다. joint1 부터 joint6 는 차례대로 base, shoulder, elbow, wrist1, wrist2, wrist3 관절의 목표 관절의 절대 각도 값입니다. degree 단위의 실수형 상수를 입력합니다. spd 와 acc 는 각각 속도와 가속도 값을 나타냅니다. 0~1 사이의 실수형 상수를 입력합니다. 0 에 가까울수록 느린 값을 나타냅니다. -1 을 입력할 경우에는 시스템에 설정된 기본 속도와 가속도 값으로 움직입니다. 기존에 다른 동작 명령을 수행 중이었다면 이 명령은 무시됩니다.
예시	"jointall 0.4, 0.1, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0"

movetcp

명령어	movetcp
스크립트	movetcp spd, acc, x, y, z, rx, ry, rz
설명	이 명령은 입력된 TCP 의 위치와 회전각을 목표로 Cartesian Space 에서 움직입니다. x, y, z 는 TCP 의 위치를 나타내는 값으로 mm 단위의 실수형 상수입니다. rx, ry, rz 는 TCP 의 방향을 결정하기 위한 Yaw-Pitch-Roll 순서의 오일러 각도를 나타내고, degree 단위의 실수형 상수를 입력합니다. spd 와 acc 는 각각 속도와 가속도 값을 나타냅니다. 0~1 사이의 실수형 상수를 입력합니다. 0 에 가까울수록 느린 값을 나타냅니다. -1 을 입력할 경우에는 시스템에 설정된 기본 속도와 가속도 값으로 움직입니다. 기존에 다른 동작 명령을 수행 중이었다면 이 명령은 무시됩니다.
예시	“movetcp 0.4, 0.1, 100.0, 100.0, 300.0, 0.0, 90.0, 0.0”

movecircle

명령어	movecircle(three points 모드)
스크립트	movecircle threepoints orientation_option spd, acc, x1, y1, z1, rx1, ry1, rz1, x2, y2, z2, rx2, ry2, rz2
설명	이 명령은 TCP 의 현재 위치를 시작으로 입력된 중간 위치와 최종 위치를 이용해 Cartesian Space 에서 원을 그리는 동작입니다. orientation_option 을 통해서 원을 그릴 때 TCP 의 방향을 결정하는 3 가지 옵션을 선택할 수 있습니다. “intended”는 중간 위치와 최종 위치에 입력한 방향 (rx1, ry1, rz1 과 rx2, ry2, rz2)을 모두 지키면서 원을 그립니다. “constant”는 시작점인 TCP 의 현재 방향을 유지한 채로 원을 그립니다. “radial”은 그리게 되는 원의 중심방향으로 TCP 의 방향이 상대적으로 변합니다. 방향의 기준은 현재 TCP 의 방향입니다. x1, y1, z1 는 원의 중간 위치에서의 TCP 의 위치를 나타내는 값으로 mm 단위의 실수형 상수입니다.

	rx1, ry1, rz1 는 원의 중간 위치에서의 TCP 의 방향을 결정하기 위한 Yaw-Pitch-Roll 순서의 오일러 각도를 나타내고, degree 단위의 실수형 상수를 입력합니다. x2, y2, z2 는 원의 최종 위치에서의 TCP 의 위치를 나타내는 값으로 mm 단위의 실수형 상수입니다. rx2, ry2, rz2 는 원의 최종 위치에서의 TCP 의 방향을 결정하기 위한 Yaw-Pitch-Roll 순서의 오일러 각도를 나타내고, degree 단위의 실수형 상수를 입력합니다. spd 와 acc 는 각각 속도와 가속도 값을 나타냅니다. 0~1 사이의 실수형 상수를 입력합니다. 0 에 가까울수록 느린 값을 나타냅니다. -1 을 입력할 경우에는 시스템에 설정된 기본 속도와 가속도 값으로 움직입니다. 기존에 다른 동작 명령을 수행 중이었다면 이 명령은 무시됩니다.
예시	"movecircle threepoints intended 0.4, 0.1, 100.0, 100.0, 300.0, 0.0, 90.0, 0.0, 200.0, 200.0, 200.0, 0.0, 90.0, 45.0" "movecircle threepoints constant 0.4, 0.1, 100.0, 100.0, 300.0, 0.0, 90.0, 0.0, 200.0, 200.0, 200.0, 0.0, 90.0, 45.0" "movecircle threepoints radial 0.4, 0.1, 100.0, 100.0, 300.0, 0.0, 90.0, 0.0, 200.0, 200.0, 200.0, 0.0, 90.0, 45.0"

명령어	movecircle(axis 모드)
스크립트	movecircle axis orientation_option spd, acc, rot_angle, cx, cy, cz, ax, ay, az
설명	이 명령은 TCP 의 현재 위치에서 시작해서 입력된 회전축을 중심으로 입력된 각도만큼 원을 그리는 동작입니다. orientation_option 을 통해서 원을 그릴 때 TCP 의 방향을 결정하는 3 가지 옵션을 선택할 수 있습니다. "intended"는 아래의 "constant"와 동일합니다. "constant"는 시작점인 TCP 의 현재 방향을 유지한 채로 원을 그립니다. "radial"은 그리게 되는 원의 중심방향으로 TCP 의 방향이 상대적으로 변합니다. 방향의 기준은 현재 TCP 의 방향입니다. cx, cy, cz 는 회전 중심점의 위치를 나타내는 값으로 mm 단위의 실수형 상수입니다. ax, ay, az 는 회전의 축을 나타내는 벡터로 실수형 상수를 입력합니다. rot_angle 은 회전할 양을 나타내는 값으로 degree 단위의 실수형 상수입니다.

	spd 와 acc 는 각각 속도와 가속도 값을 나타냅니다. 0~1 사이의 실수형 상수를 입력합니다. 0 에 가까울수록 느린 값을 나타냅니다. -1 을 입력할 경우에는 시스템에 설정된 기본 속도와 가속도 값으로 움직입니다. 기존에 다른 동작 명령을 수행 중이었다면 이 명령은 무시됩니다.
예시	"movecircle axis constant 0.4, 0.1, 180.0, 200.0, 200.0, 200.0, 1.0, 0.0, 0.0" "movecircle axis radial 0.4, 0.1, 180.0, 200.0, 200.0, 200.0, 1.0, 0.0, 0.0"

blend_jnt

명령어	blend_jnt
스크립트	blend_jnt clear_pt
설명	이 명령은 Joint Space 에서 연속된 목적 관절 각도들을 연결하는 동작 명령 중에서 저장된 목적 값들을 지우는 명령어입니다. 새로운 연속 관절 각도들을 입력하기 전의 사전 동작으로 사용합니다.
예시	"blend_jnt clear_pt"

명령어	blend_jnt
스크립트	blend_jnt add_pt spd, acc, joint1, joint2, joint3, joint4, joint5, joint6
설명	이 명령은 Joint Space 에서 연속된 목적 관절 각도들을 연결하는 동작 명령 중에서 목적 관절 각도들을 입력하는 명령어입니다. joint1 부터 joint6 는 차례대로 base, shoulder, elbow, wrist1, wrist2, wrist3 관절의 목표 관절의 절대 각도 값입니다. degree 단위의 실수형 상수를 입력합니다. spd 와 acc 는 각각 속도와 가속도 값을 나타냅니다. 0~1 사이의 실수형 상수를 입력합니다. 0 에 가까울수록 느린 값을 나타냅니다. -1 을 입력할 경우에는 시스템에 설정된 기본 속도와 가속도 값으로 움직입니다. 마지막으로 입력된 spd 와 acc 로 전체 "blend_jnt" 명령의 속도와 가속도가 결정됩니다.
예시	"blend_jnt add_pt 0.4, 0.1, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0"

명령어	blend_jnt
스크립트	blend_jnt move_pt

설명	이 명령은 Joint Space 에서 연속된 목적 관절 각도들을 연결하는 동작 명령 중에서 동작을 시작시키는 명령어입니다. 기존에 "blend_jnt add_pt"로 입력한 관절 값들로 차례대로 움직이는 동작이 수행됩니다.
예시	"blend_jnt move_pt"

blend_tcp

명령어	blend_tcp
스크립트	blend_tcp clear_pt
설명	이 명령은 Cartesian Space 에서 연속된 목적 위치들을 연결하는 동작 명령 중에서 저장된 목적 값들을 지우는 명령어입니다. 새로운 연속 목적 위치들을 입력하기 전의 사전 동작으로 사용합니다.
예시	"blend_tcp clear_pt"

명령어	blend_tcp
스크립트	blend_tcp add_pt spd, acc, radius, x, y, z, rx, ry, rz
설명	이 명령은 Cartesian Space 에서 연속된 목적 위치들을 연결하는 동작 명령 중에서 목적 위치들을 입력하는 명령어입니다. radius 는 목적 위치들을 연달아 이동할 때 곡선을 그리며 다음 목적 위치로 이동하기 위한 거리를 나타낸 값으로 mm 단위의 상수를 입력합니다. 하나의 목적 위치에서 다음 목적 위치로의 이동을 선분으로 생각할 때 그 절반 값이 radius 가 가질 수 있는 최대 값입니다. 이 값보다 크게 주면 내부적으로 알아서 값을 줄여서 사용합니다. x, y, z 는 TCP 의 위치를 나타내는 값으로 mm 단위의 실수형 상수입니다. rx, ry, rz 는 TCP 의 방향을 결정하기 위한 Yaw-Pitch-Roll 순서의 오일러 각도를 나타내고, degree 단위의 실수형 상수를 입력합니다. spd 와 acc 는 각각 속도와 가속도 값을 나타냅니다. 0~1 사이의 실수형 상수를 입력합니다. 0 에 가까울수록 느린 값을 나타냅니다. -1 을 입력할 경우에는 시스템에 설정된 기본 속도와 가속도 값으로 움직입니다.

	마지막으로 입력된 spd 와 acc 로 전체 “blend_tcp” 명령의 속도와 가속도가 결정됩니다.
예시	“blend_tcp add_pt 0.4, 0.1, 30.0, 100.0, 100.0, 300.0, 0.0, 90.0, 0.0”

명령어	blend_tcp
스크립트	blend_tcp move_pt
설명	이 명령은 Cartesian Space 에서 연속된 목적 위치들을 연결하는 동작 명령 중에서 동작을 시작시키는 명령어입니다. 기존에 “blend_tcp add_pt”로 입력한 목적 위치들로 차례대로 움직이는 동작이 수행됩니다.
예시	“blend_tcp move_pt”

다음으로 설명드릴 스크립트는 배전반과 툴플랜지의 디지털과 아날로그 포트의 출력 값을 제어하는 명령입니다. 명령어는 3 가지입니다.

1) digital_out
2) analog_out
3) tool_out

digital_out

스크립트	digital_out d0, d1, d2, d3, d4, d5, d6, d7, d8, d9, d10, d11, d12, d13, d14, d15
설명	이 명령은 제어반의 디지털 출력 포트를 통해서 신호를 내보냅니다. d0~d15 는 각각의 디지털 포트의 on/off 를 나타냅니다. 값이 1 이면 on 이고, 0 이면 off 입니다. d0~d15 에 0 과 1 이외에 -1 을 입력할 수 있습니다. 이 경우에는 해당 포트의 기존 출력 값을 유지합니다.
예시	“digital_out 1, 1, 1, 1, 0, 0, 0, 0, -1, -1, -1, -1, -1, -1, -1, -1”

analog_out

스크립트	analog_out a0, a1, a2, a3
설명	이 명령은 제어반의 아날로그 출력 포트를 통해서 신호를 내보냅니다. a0~a3 은 각각의 아날로그 포트의 출력 값을 나타냅니다. 값이 0 에서 10 의 값을 가지며 voltage 단위입니다. a0~a3 에 -1 을 입력할 수 있습니다. 이 경우에는 해당 포트의 기존 출력 값을 유지합니다.
예시	"analog_out 5.0, 5.0, -1, -1"

tool_out

스크립트	tool_out volt, d0, d1
설명	이 명령은 툴플랜지의 전압 레벨을 설정하고, 디지털 출력 포트를 통해서 신호를 내보냅니다. volt 는 툴플랜지의 디지털 출력의 전압 레벨을 설정합니다. 값은 0, 12, 24 가 가능하고, 이는 voltage 단위입니다. 이 세개의 값 이외의 입력은 무시됩니다. volt 에 -1 을 입력할 수 있습니다. 이 경우에는 기존의 전압 설정을 유지합니다. d0, d1 은 각각의 디지털 포트의 on/off 를 나타냅니다. 값이 1 이면 on 이고, 0 이면 off 입니다. d0, d1 에 0 과 1 이외에 -1 을 입력할 수 있습니다. 이 경우에는 해당 포트의 기존 출력 값을 유지합니다.
예시	"tool_out 12, 1, 0"

다음으로 설명드릴 스크립트는 초기화, 종료, 동작모드 변경, 속도 변경입니다.

1) mc
2) shutdown
3) pgmode
4) sdw

mc

스크립트	mc jall init
설명	이 명령은 로봇 하드웨어를 초기화합니다.
예시	“mc jall init”

shutdown

스크립트	shutdown
설명	이 명령은 로봇 동작을 종료시키고, 전원을 내립니다.
예시	“shutdown”

pgmode

스크립트	pgmode mode_type
설명	이 명령은 로봇의 동작 모드를 변경합니다. mode_type 을 통해서 동작 모드를 선택합니다. 동작 모드에는 “real”과 “simulation”이 있습니다. “real”에서는 동작 명령을 내릴 때 실제로 로봇이 움직입니다. “simulation”에서는 동작 명령을 내릴 때 내부 레퍼런스만 변경되고, 그 명령이 로봇을 움직이지는 않습니다. 로봇을 처음 초기화 하였을 때 기본 모드는 “simulation” 모드입니다.
예시	“pgmode real” “pgmode simulation”

sdw (shared data write)

스크립트	sdw default_speed spd
설명	이 명령은 로봇의 모든 동작에 대한 기본 속도 배율을 지정합니다. spd 에는 0~1 사이의 실수형 상수를 입력합니다. 0 에 가까울수록 느린 값을 나타냅니다. 이 값이 0 이면 로봇은 움직이지 않습니다. 또한 레퍼런스도 변하지 않습니다. 이 속도 값은 팬던트가 연결되어 있을 경우에, 특히 “Make” 모드에 진입한 상태일 경우에 팬던트에 의해서도 값이 제어가 되기 때문에 사용에 유의하시기 바랍니다. 외부 제어와 팬던트 사이에 별도의 우선권이 존재하지 않고, 마지막에 입력한 값이 최종 값입니다.
예시	“sdw default_speed 0.5”

마지막으로 설명드릴 스크립트는 task 스크립트입니다.

task

스크립트	task load work_file_name
설명	이 명령은 기존에 작성된 워크 파일을 로드합니다. work_file_name 에는 “.wsl” 파일의 이름이 사용됩니다. 이때에 “.wsl”은 생략하고, 파일의 상대경로와 이름만 입력합니다. 이 파일은 팬던트에 존재하는 파일이 아니라, 팬던트를 통해서 한번이라도 제어반에 연결하여 로드 혹은 저장했었던 파일이어야 합니다. 그렇기 때문에 팬던트가 연결되어 있지 않더라도 해당 파일을 로드할 수 있습니다.
예시	“task load test_file”

스크립트	task play option
설명	이 명령은 로드된 워크 파일을 실행합니다. option 에는 아무런 값이 들어가지 않거나 "once"가 입력될 수 있습니다. option 에 아무런 값을 주지 않으면 설정된 횟수만큼 워크파일을 실행합니다. option 에 "once"를 입력할 경우에 워크 파일은 단 한 번만 실행됩니다.
예시	"task play" "task play once"

스크립트	task repeat num
설명	이 명령은 워크 파일을 실행할 경우의 반복 횟수를 지정합니다. num 에는 반복 횟수를 정수형으로 입력합니다. 만약 num 에 -1 을 입력하면 무한대를 의미합니다. 이 명령으로 설정한 반복 횟수는 배전반의 전원을 리붓 시키기 전까지 유지됩니다. 그 이후에는 팬던트로 설정한 값으로 다시 치환됩니다.
예시	"task repeat 5" "task repeat -1"

스크립트	task pause
설명	이 명령은 실행 중이던 로봇 동작을 일시정지 시킵니다. 외부 제어를 통한 동작 명령과 "task play" 명령으로 인해 실행 중이던 워크 파일에 모두 사용할 수 있습니다. 일시정지된 상태에서는 "task resume_a" 명령을 통해서 동작을 재개할 수 있습니다. 로봇이 일시정지된 상태일 때에는 동작이 끝난 것으로 생각하지 않기 때문에 외부 제어에서 다른 동작 명령을 주어도 무시됩니다.
예시	"task pause"

스크립트	task stop
설명	이 명령은 실행 중이던 로봇 동작을 정지시킵니다. 외부 제어를 통한 동작 명령과 “task play” 명령으로 인해 실행 중이던 워크 파일에 모두 사용할 수 있습니다. 동작이 정지된 상태에서는 “task resume_a”로 동작이 재개되지 않습니다. 동작이 완전 종료된 상태입니다. 동작을 정지 명령을 내리면 로봇이 빠른 동작을 하고 있던 중에 급격히 멈출 수 있습니다. “task pause” 이후에 “task stop” 명령을 수행하기를 권장합니다.
예시	“task stop”

스크립트	task resume_a
설명	이 명령은 “task pause”나 워크 파일에서 alarm 혹은 debug 명령으로 일시정지된로봇의 동작을 재개합니다.
예시	“task resume_a”

스크립트	task resume_b
설명	이 명령은 외부 충돌 감지로 인해 일시정지된 로봇의 동작을 재개합니다.
예시	“task resume_b”

외부 제어를 사용하기 위해서 제어반의 프로그램과 연결을 해야 합니다. 연결은 TCP/IP 통신을 사용하고, 해당하는 IP 주소는 팬던트에서 설정할 수 있습니다. 그리고 그 결과는 제어반의 LCD 화면에 표시됩니다. 외부 제어를 위해 5000 번과 5001 번 포트가 열립니다. 5000 번 포트는 명령어를 받기 위한 포트이고, 5001 번 포트는 로봇 상태를 나타내는 데이터를 요청받고 보내주는 역할을 위한 포트입니다. 편의상 5000 번 포트는 명령 포트라고 하고, 5001 번 포트를 데이터 포트라고 하겠습니다.

명령 포트로는 앞에서 설명한 스크립트 명령어를 전송하면 됩니다. 명령 포트에는 첫 명령어에 대한 필터가 있어서 시작이 “jointall”, “movetcp”, “mc”, “pgmode”, 등과 같은 앞서 설명한 스크립트 명령어가 아닐 경우에는 “The command is not allowed”라는 문구를 응답으로 보냅니다. 정상적인 명령어로 시작되어 파서로 입력 문장을 넘겼을 경우에는 “The command was executed”라는 문구의 응답이 옵니다.

데이터 포트로는 "reqdata" 라는 명령어를 보내면 그에 대한 응답으로 현대 로봇 상태에 대한 정보가 데이터 포트로 들어옵니다. 데이터의 형식은 아래와 같습니다.

Header (4 Byte)				Data (n Byte)
0x24	Size&0xFF	(Size>>8)&0xFF	0x03	Data

그리고 Data 의 형식은 아래와 같습니다. 시스템의 버전에 따라서 Data 의 사이즈가 다를 수도 있습니다. 하지만 순서는 일치하니 아래 표를 참조하시기 바랍니다.

Offset	타입	설명
0	Float	Task 가 수행된 시간(초단위 – 태스크가 시작할 때 리셋)
1	Float	현재 base 관절의 레퍼런스 각도(degree 단위)
2	Float	현재 shoulder 관절의 레퍼런스 각도(degree 단위)
3	Float	현재 elbow 관절의 레퍼런스 각도(degree 단위)
4	Float	현재 wrist1 관절의 레퍼런스 각도(degree 단위)
5	Float	현재 wrist2 관절의 레퍼런스 각도(degree 단위)
6	Float	현재 wrist3 관절의 레퍼런스 각도(degree 단위)
7	Float	현재 base 관절의 엔코더 각도(degree 단위)
8	Float	현재 shoulder 관절의 엔코더 각도(degree 단위)
9	Float	현재 elbow 관절의 엔코더 각도(degree 단위)
10	Float	현재 wrist1 관절의 엔코더 각도(degree 단위)
11	Float	현재 wrist2 관절의 엔코더 각도(degree 단위)
12	Float	현재 wrist3 관절의 엔코더 각도(degree 단위)
13	Float	현재 base 관절의 전류값(Ampere 단위)
14	Float	현재 shoulder 관절의 전류값(Ampere 단위)
15	Float	현재 elbow 관절의 전류값(Ampere 단위)
16	Float	현재 wrist1 관절의 전류값(Ampere 단위)
17	Float	현재 wrist2 관절의 전류값(Ampere 단위)
18	Float	현재 wrist3 관절의 전류값(Ampere 단위)
19	Float	현재 TCP 레퍼런스의 위치 X 값(mm 단위)
20	Float	현재 TCP 레퍼런스의 위치 Y 값(mm 단위)
21	Float	현재 TCP 레퍼런스의 위치 Z 값(mm 단위)
22	Float	현재 TCP 레퍼런스의 방향 각도 RX 값(degree 단위)

23	Float	현재 TCP 레퍼런스의 방향 각도 RY 값(degree 단위)
24	Float	현재 TCP 레퍼런스의 방향 각도 RZ 값(degree 단위)
25	Float	19 와 동일
26	Float	20 와 동일
27	Float	21 와 동일
28	Float	22 와 동일
29	Float	23 와 동일
30	Float	24 와 동일
31	Float	제어반 아날로그 신호 0 번 포트 입력값(voltage 단위)
32	Float	제어반 아날로그 신호 1 번 포트 입력값(voltage 단위)
33	Float	제어반 아날로그 신호 2 번 포트 입력값(voltage 단위)
34	Float	제어반 아날로그 신호 3 번 포트 입력값(voltage 단위)
35	Float	제어반 아날로그 신호 0 번 포트 출력값(voltage 단위)
36	Float	제어반 아날로그 신호 1 번 포트 출력값(voltage 단위)
37	Float	제어반 아날로그 신호 2 번 포트 출력값(voltage 단위)
38	Float	제어반 아날로그 신호 3 번 포트 출력값(voltage 단위)
39	Int	제어반 디지털 신호 0 번 포트 입력값(on:1 / off:0)
40	Int	제어반 디지털 신호 1 번 포트 입력값(on:1 / off:0)
41	Int	제어반 디지털 신호 2 번 포트 입력값(on:1 / off:0)
42	Int	제어반 디지털 신호 3 번 포트 입력값(on:1 / off:0)
43	Int	제어반 디지털 신호 4 번 포트 입력값(on:1 / off:0)
44	Int	제어반 디지털 신호 5 번 포트 입력값(on:1 / off:0)
45	Int	제어반 디지털 신호 6 번 포트 입력값(on:1 / off:0)
46	Int	제어반 디지털 신호 7 번 포트 입력값(on:1 / off:0)
47	Int	제어반 디지털 신호 8 번 포트 입력값(on:1 / off:0)
48	Int	제어반 디지털 신호 9 번 포트 입력값(on:1 / off:0)
49	Int	제어반 디지털 신호 10 번 포트 입력값(on:1 / off:0)
50	Int	제어반 디지털 신호 11 번 포트 입력값(on:1 / off:0)
51	Int	제어반 디지털 신호 12 번 포트 입력값(on:1 / off:0)
52	Int	제어반 디지털 신호 13 번 포트 입력값(on:1 / off:0)
53	Int	제어반 디지털 신호 14 번 포트 입력값(on:1 / off:0)
54	Int	제어반 디지털 신호 15 번 포트 입력값(on:1 / off:0)
55	Int	제어반 디지털 신호 0 번 포트 출력값(on:1 / off:0)
56	Int	제어반 디지털 신호 !번 포트 출력값(on:1 / off:0)
57	Int	제어반 디지털 신호 2 번 포트 출력값(on:1 / off:0)

58	Int	제어반 디지털 신호 3 번 포트 출력값(on:1 / off:0)
59	Int	제어반 디지털 신호 4 번 포트 출력값(on:1 / off:0)
60	Int	제어반 디지털 신호 5 번 포트 출력값(on:1 / off:0)
61	Int	제어반 디지털 신호 6 번 포트 출력값(on:1 / off:0)
62	Int	제어반 디지털 신호 7 번 포트 출력값(on:1 / off:0)
63	Int	제어반 디지털 신호 8 번 포트 출력값(on:1 / off:0)
64	Int	제어반 디지털 신호 9 번 포트 출력값(on:1 / off:0)
65	Int	제어반 디지털 신호 10 번 포트 출력값(on:1 / off:0)
66	Int	제어반 디지털 신호 11 번 포트 출력값(on:1 / off:0)
67	Int	제어반 디지털 신호 12 번 포트 출력값(on:1 / off:0)
68	Int	제어반 디지털 신호 13 번 포트 출력값(on:1 / off:0)
69	Int	제어반 디지털 신호 14 번 포트 출력값(on:1 / off:0)
70	Int	제어반 디지털 신호 15 번 포트 출력값(on:1 / off:0)
71	Float	현재 base 관절의 모터제어기 온도(celcius 단위)
72	Float	현재 shoulder 관절의 모터제어기 온도(celcius 단위)
73	Float	현재 elbow 관절의 모터제어기 온도(celcius 단위)
74	Float	현재 wrist1 관절의 모터제어기 온도(celcius 단위)
75	Float	현재 wrist2 관절의 모터제어기 온도(celcius 단위)
76	Float	현재 wrist3 관절의 모터제어기 온도(celcius 단위)
77	Int	태스크의 프로그램 카운터 위치(Step 명령이 수행될 위치)
78	Int	태스크 목표 반복 횟수
79	Int	현재 수행 중이 태스크의 액션 번호
80	Int	태스크의 현재 반복 횟수
81	Float	태스크 수행 시간(초단위 – 태스크가 시작할 때 리셋 안됨)
82	Int	태스크 상태(1: Idle, 2: Paused, 3: Run)
83	Float	로봇 동작의 기본 속도 배율(0~1)
84	Float	로봇 동작 상태(1: 정지, 3: 동작중)
85	Float	파워 상태 - LSB offset 기준으로 0: 48V 입력 1: 48V 출력 2: 24V 상태 3: 비상스위치 상태 4: PC 스위치 상태 5: 모터 제어기 상태
86	Float	Not used
87	Float	Not used

88	Float	Not used
89	Float	Not used
90	Float	Not used
91	Float	Not used
92	Int	현재 base 관절의 모터제어기 상태
93	Int	현재 shoulder 관절의 모터제어기 상태
94	Int	현재 elbow 관절의 모터제어기 상태
95	Int	현재 wrist1 관절의 모터제어기 상태
96	Int	현재 wrist2 관절의 모터제어기 상태
97	Int	현재 wrist3 관절의 모터제어기 상태
		모터제어기 상태 정보 – LSB offset 기준으로 0: FET 상태 1: 위치제어 상태 2: 초기화 결과 3: 제어 모드 4: Nonius 에러 5: Low Battery 6: 캘리브레이션 모드 7: Multiturn 에러 8: JAM 에러 9: Over Current 에러 10: Big 에러 11: Input 에러 12: FET 드라이버 에러 13: 온도 에러 14: 위치 에러(Low) 15: 위치 에러(High)
98	Int	충돌 감지 on-off(1: on, 0: off)
99	Int	직접 교시 상태(1: on, 0: off)
100	Int	로봇 동작 모드(1: simulation mode, 0: real mode)
101	Int	로봇 초기화 상태 정보 0: 기본값 1: 전압 체크 2: 디바이스 체크 3: 로봇 위치제어 시작 4: 변수 체크 5: 충돌 체크

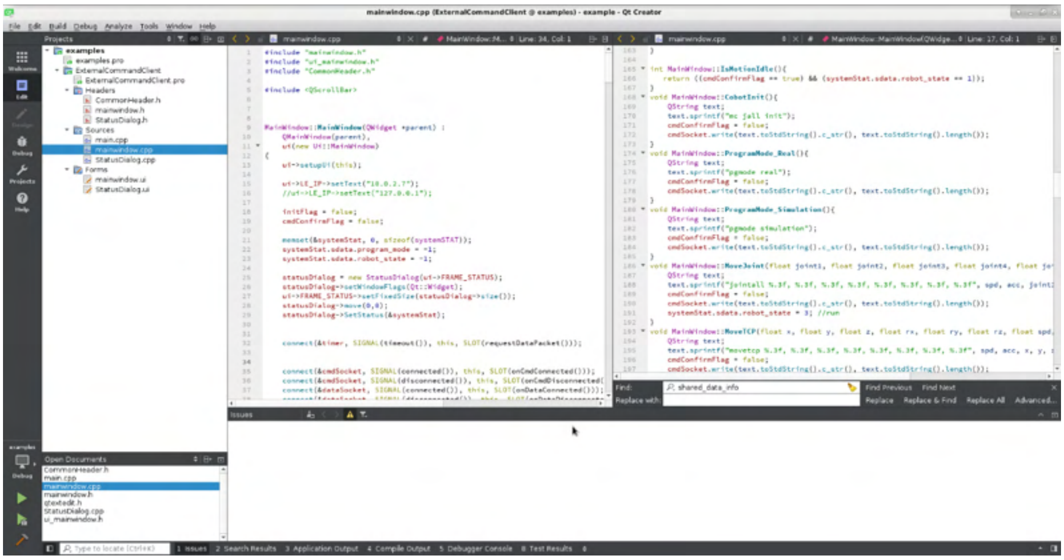
		6: 초기화 완료
102	Int	로봇 초기화 에러 코드 0: 에러 없음 1: SMPS 에러 2: 비상 스위치 에러 3: 컨트롤 박스 내부 파워 전환 에러 1 4: 컨트롤 박스 내부 파워 전환 에러 2 5: 로봇 접속 에러 6: 로봇 초기화 에러 7: 부하 세팅 검사 에러 8: 툴플랜지 접속 에러 9: 툴플랜지 각도 에러 10: 모터 제어기 엔코더 초기값 에러 1 11: 모터 제어기 엔코더 초기값 에러 2 12: 디지털 인풋 16/17 단락 에러 13: 48V 스위치 에러 14: 직접 교시 버튼 눌림 에러
103	Float	툴플랜지 아날로그 신호 0 번 포트 입력값(voltage 단위)
104	Float	툴플랜지 아날로그 신호 1 번 포트 입력값(voltage 단위)
105	Int	툴플랜지 디지털 신호 0 번 포트 입력값(on:1 / off:0)
106	Int	툴플랜지 디지털 신호 1 번 포트 입력값(on:1 / off:0)
107	Int	툴플랜지 디지털 신호 0 번 포트 출력값(on:1 / off:0)
108	Int	툴플랜지 디지털 신호 1 번 포트 출력값(on:1 / off:0)
109	Float	툴플랜지의 출력 전압 레벨(voltage 단위)
110	Int	외부 충돌 감지 상태(on:1 / off:0)
111	Int	로봇 장치 오류 발견 상태 0: 오류 없음 1: PVL 에러 2: CPU 에러 3: Big 에러 4: Input 에러 5: JAM 에러 6: Over Current 에러 7: 관절 각도 에러 8: 제어 모드 에러 9: 레퍼런스 & 엔코더 오차 에러

		10: 상위제어기 전류 에러 11: 온도 에러 12: 직접 교시 속도 초과 에러
112	Int	자가 충돌 감지 상태(on:1 / off:0)
113	Int	로봇 일시정지 상태(on:1 / off:0)
114	Int	로봇 동작 오류 발견 상태 0: 오류 없음 1: 로봇이 쭉 뻗은 상태에서 TCP 명령 들어옴 2: TCP 명령의 한계 도달 3: 관절 각도 명령의 한계 도달 4: 해가 없는 TCP 명령
115	Int	제어반 디지털 신호 16 번 포트 입력값(on:1 / off:0)
116	Int	제어반 디지털 신호 17 번 포트 입력값(on:1 / off:0)
117	Int	Inbox 0 Trap 발생
118	Int	Inbox 1 Trap 발생
119	Int	Inbox 0 체크 모드
120	Int	Inbox 1 체크 모드

E.2 예제 프로그램 개발환경

본 예제는 Debian 9.8 버전과 Ubuntu 18.04 에서 테스트되었습니다. 유사한 계열의 Linux 시스템에서도 동작 가능할 수 있습니다. 별도의 커널 패치는 필요하지 않습니다.

프로그래밍을 작성하는 IDE(Integrated Development Environment)로는 Qt 5.8 버전 (https://www.qt.io)를 사용합니다.



주의

주의:

Qt 기반의 C++ 예제, Visual Studio 기반의 C# 예제 등이 있습니다. 예제 프로그램은 제조사 또는 대리점을 통해 받을 수 있습니다.

E.3 프로그래밍 방법

본 예제에는 태블릿 UI(User Interface)가 제공하는 모든 기능이 포함된 것이 아닙니다. 외부 제어를 통해 로봇을 움직이는 동안에 사용자가 모니터링하면 유용한 정보들에 대해서만 구현되어 있습니다.

예제를 실행시켰을 때 다음 그림과 같은 UI 가 실행됩니다.

각 항목별 기능은 아래와 같습니다.

1) 네트워크 연결

RB Series 로봇의 컨트롤 박스에 있는 랜(LAN) 포트를 통해서 로봇의 메인 제어기에 접속합니다. 외부 제어를 위한 기본 IP 주소는 '10.0.2.7'로 고정되어 있습니다. 외부 제어 명령어를 받기 위한 서버는 포트 5000 으로 접속하고, 로봇의 상태 정보를 요청하고 받기 위한 서버는 포트 5001 로 접속합니다. 각각을 접속하기 위한 버튼은 따로 존재하고, 접속에 성공하면 버튼에 적혀 있는 'Connect'라는 단어가 'Disconnect'로 변경됩니다. 접속이 끊겼을 때에는 반대로 됩니다.

2) 로봇 초기화

로봇의 메인 제어기에 접속이 된 상태에서 'Cobot Init'이 적혀 있는 버튼을 누르면 로봇이 초기화 과정을 수행합니다. 로봇은 'Power Set', 'Device Set', 'System Set', 그리고 'Robot Init'이라는 과정을 순차적으로 거칩니다. 각 과정을 수행 중일때에는 각 과정의 앞에 있는 흰색 에디트 박스가 노란색으로 바뀝니다. 수행이 완료된 과정은 녹색으로 바뀌고, 아직 수행하지 못한 과정은 빨간색으로 유지됩니다. 모든 4 개의 박스가 초록색으로 바뀌면 로봇의 초기화 과정이 완료된 상태로, 동작을 수행할 준비가 됩니다.

3) 로봇 상태

로봇의 상태는 컨트롤 박스에 있는 메인 프로그램으로부터 전달받는 데이터로부터 알 수 있습니다. 이 데이터는 포트 5001 로 'reqdata'를 요청하면 메인 프로그램에서 응답으로 보내줍니다. 데이터의 형식은 'CommonHeader.h'의 'systemSTAT' 구조체의 형태로 전달됩니다.

run mode: 로봇의 동작 모드를 표시합니다. 리얼 모드와 시뮬레이션 모드가 있습니다. 리얼 모드에서는 동작 명령이 실제로 로봇에 인가되어 로봇이 움직입니다. 시뮬레이션 모드에서는 동작을 수행하기는 하지만 그 명령이 로봇에는 전달되지 않습니다. 태블릿이 연결되어 있다면 반투명한 로봇이 움직이는 모습을 볼 수 있습니다. 로봇 동작 모드는 'systemSTAT'의 'program_mode' 변수 값으로 대변됩니다. 이 변수의 값이 0 이면 리얼 모드이고, 1 이면 시뮬레이션 모드입니다.

robot state: 로봇이 현재 움직이는 중인지 아니면 동작 명령을 받을 수 있는 상태인지 표시합니다. 로봇상태는 'systemSTAT'의 'robot_state' 변수값으로 부터 알 수 있습니다. 만약 이 값이 1 이면 동작 명령을 받을 수 있는 idle 상태이고, 3 이면 로봇이 움직이고 있는 상태입니다. 로봇이 움직이고 있는 동안에는 동작 명령어는 무시됩니다. 만약 'robot_state'값이 2 이면 로봇이 불특정한 이유로 동작이 중단된 상태이거나 일시정지 명령을 통해 멈춰 있는 상태입니다. 이때에는 'status'항목의 paused 로 표시됩니다.

status: 현재 로봇의 특수 동작 상태 혹은 이상 상태를 표시합니다. 직접 교시 중이면 'teaching', 외부 충돌 감지로 인해 정지된 상태면 'ext. collision', 동작 중에 자가 충돌 직전이면 'self collision', 일시정지 명령에 의해 멈춘 상태면 'paused', 로봇 제어 알고리즘 상의 솔루션이 없는 입력이 들어오면 'ems', 전원 문제나 로봇 제어에 문제가 발생하면 'sos' 에디트 창의 색이 변합니다. 'systemSTAT'의 'op_stat_collision_occur', 'op_stat_sos_flag', 'op_stat_self_collision', 'op_stat_soft_estop_occur', 'op_stat_ems_flag' 와 'robot_state' 값을 참조하면 표시됩니다.

joint reference: 각 관절의 레퍼런스 입력 값을 표시합니다(degree 단위).

joint encoder: 각 관절의 현재 엔코더 값을 표시합니다(degree 단위).

TCP reference: TCP 의 레퍼런스 위치 값을 표시합니다(mm 와 degree 단위).

digital in: 컨트롤 박스의 디지털 입력 값을 표시합니다.

digital out: 컨트롤 박스의 디지털 출력 값을 표시합니다.

analog in: 컨트롤 박스의 아날로그 입력 값을 표시합니다(voltage 단위).

analog out: 컨트롤 박스의 아날로그 출력 값을 표시합니다(voltage 단위).

tool out voltage: 현재 설정된 툴 플랜지 보드의 출력 전압을 표시합니다(0V, 12V, 24V 중 택일 1).

tool digital in: 툴 플랜지 보드의 디지털 입력 값을 표시합니다.

tool digital out: 툴 플랜지 보드의 디지털 출력 값을 표시합니다.

tool analog in: 툴 플랜지 보드의 아날로그 입력 값을 표시합니다.

4) 모드변환

로봇은 시뮬레이션 모드와 리얼 모드의 두가지 동작 모드를 가질 수 있습니다. 시뮬레이션 모드에서는 로봇은 움직이지 않지만 입력 레퍼런스의 값은 변경할 수 있습니다. 리얼 모드에서는 사용자의 입력에 따라 로봇이 실제로 움직입니다. 'Real'과 'Simulation'이 적힌 버튼을 누름에 따라서 로봇의 동작 모드를 변환할 수 있습니다. 로봇의 초기화 과정을 마친 직후에는 시뮬레이션 모드입니다.

5) 속도조절

로봇 동작의 전체적인 속도를 조절합니다. 슬라이더바를 옮겨서 0%에서 100%사이에서 조절할 수 있습니다. 이 속도는 로봇의 동작 명령에 주어지는 속도에 곱해져서 작용합니다.

6) 모션 정지 및 재개

로봇이 동작하는 중간에 일시정지를 하거나 멈출 수 있습니다. 일시정지를 하기 위해서는 'Motion Pause' 버튼을 누르고, 모션을 완전히 멈추기 위해서는 'Motion Halt'를 누릅니다. 'Motion Halt'의 경우에는 로봇이 급정지 하기 때문에 로봇을 안정적으로 사용하기 위해서는

일시정지를 먼저 사용하시는 것이 좋습니다. 로봇이 일시정지 상태일 경우에는 로봇에게 다른 동작 명령을 주더라도 수행되지 않습니다. 현재 동작을 멈추고 다른 동작을 하기 위해서는 'Motion Pause' 버튼 이후에 'Motion Halt' 버튼을 통해서 현재 동작을 완전히 끝내야 합니다.

반대로 일시정지 했거나 외부 충돌 감지로 인해서 멈춘 모션을 재개할 수 있습니다. 일시정지를 했던 모션을 재개하기 위해서는 'Motion Resume' 버튼을 누르고, 외부 충돌 감지로 인해 멈춘 모션을 재개하기 위해서는 'Collision Resume' 버튼을 누릅니다.

7) 디버깅 메시지 창

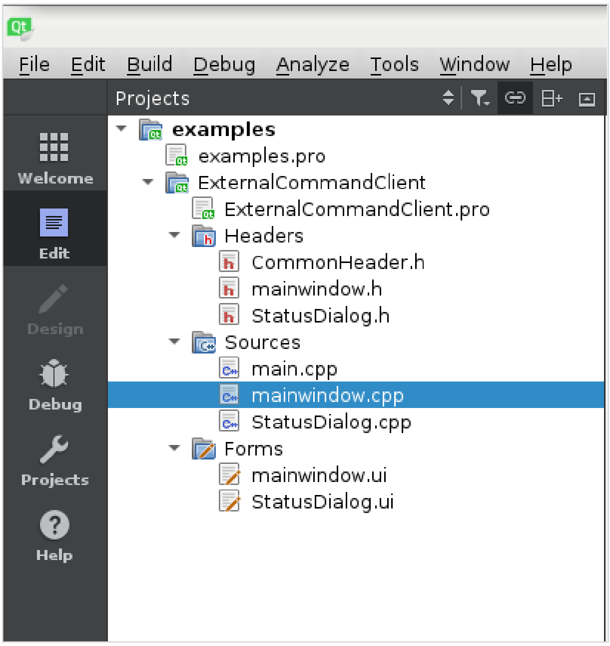
디버깅을 위한 메시지를 띄울 수 있는 창입니다.

8) 테스트 모션

'Motion Test'버튼을 누르면 기본으로 짜인 세가지 동작을 연속으로 수행합니다. 주변에 환경을 고려해서 실행하시기 바랍니다.

'Get Joint and TCP'버튼을 누르면 바로 옆의 에디트 창에 현재 로봇의 관절의 레퍼런스 각도와 TCP 값을 ',' 구분자로 표현합니다. 로봇 동작 시퀀스를 프로그램에 코딩할 때 이 값을 복사해서 사용하면 도움이 됩니다.

본 예제는 GUI 를 포함한 단일 프로세스 예제입니다. Qt 에서는 GUI 를 손쉽게 배치하고 버튼 클릭과 같은 이벤트를 생성하고 프로그램과 연결할 수 있습니다. 'mainwindow.ui'를 참조하시기 바랍니다.



예제의 핵심 내용은 'mainwindow.cpp'와 'mainwindow.h'에 포함되어 있습니다.
'CommonHeader.h'에는 로봇 상태 데이터의 형태를 확인할 수 있습니다.

```
// Cobot Control API -------------------
// <CobotInit>
// : initialize Cobot
void CobotInit();

// <MoveJoint>
// : move to target posture in joint coordinate
// joint1~joint6 : target joint angle in deg unit
// spd : speed parameter (0~1: user define   or   -1: default setting)
// acc : acceleration parameter (0~1: user define   or   -1: default setting)
void MoveJoint(float joint1, float joint2, float joint3, float joint4, float joint5, flo

// <MoveTCP>
// : move to target posture in cartesian coordinate
// x, y, z : target TCP(tool center point) position in mm unit
// rx, ry, rz : target TCP orientation (Yaw-Pitch-Roll Euler angle) in degree unit
// spd : speed parameter (0~1: user define   or   -1: default setting)
// acc : acceleration parameter (0~1: user define   or   -1: default setting)
void MoveTCP(float x, float y, float z, float rx, float ry, float rz, float spd = -1, fl

// <ControlBoxDigitalOut>
// control digital out ports in control box
// d0~d15 : digital out value (0 or 1)
void ControlBoxDigitalOut(int d0, int d1, int d2, int d3, int d4, int d5, int d6, int d7

// <ControlBoxAnalogOut>
// control analog out ports in control box
// a0~a3 : analog out value in voltage unit (0~10)
void ControlBoxAnalogOut(float a0, float a1, float a2, float a3);

// <ToolOut>
// control digital out ports and voltage level in tool flange board
// volt : reference voltage of tool flange board in voltage unit(0, 12, 24)
// d0, d1 : digital out value (0 or 1)
void ToolOut(int volt, int d0, int d1);

// <ProgramMode_Real>
// change to 'real robot' mode -- robot will move
void ProgramMode_Real();

// <ProgramMode_Simulation>
// change to 'simulation' mode -- robot will not move except teaching
void ProgramMode_Simulation();

// <BaseSpeedChange>
// change base speed -- base speed will be multiplied to motion velocity
// spd : normalized base speed (0~1)
void BaseSpeedChange(float spd);

// <MotionPause>
// pause the current motion
void MotionPause();

// <MotionHalt>
// halt the current motion
// !! CAUTION : user would better escape the motion sequence
//            : if not, the next motion will be activated immediately
void MotionHalt();

// <MotionResume>
// resume the paused motion
void MotionResume();

// <CollisionResume>
// resume the motion which is paused due to external collision
void CollisionResume();
// -----------------------------------
```

사용자가 사용할 수 있는 로봇 제어 명령어는 위와 같이 'mainwindow.h'에 명시되어 있습니다. 자세한 설명은 아래와 같습니다.

함수	CobotInit(void)
스크립트	"mc jall init"
설명	로봇의 하드웨어를 초기화합니다. 초기화 과정은 'systemSTAT'의 'init_stat_info' 와 'init_error'를 통해서 알 수 있습니다.

함수	MoveJoint(float joint1, float joint2, float joint3, float joint4, float joint5, float joint6, float spd = -1, float acc = -1);
스크립트	"jointall spd, acc, joint1, joint2, joint3, joint4, joint5, joint6"
설명	입력된 관절 값을 목표로 Joint Space 에서 움직입니다. joint1~joint6 은 목적 관절 값이고, degree 단위입니다. spd 와 acc 는 각각 속도와 가속도로 0~1 로 정규화 된 값을 가집니다. -1 을 입력할 경우에는 기본 세팅 된 속도와 가속도가 이용됩니다.

함수	MoveTCP(float x, float y, float z, float rx, float ry, float rz, float spd = -1, float acc = -1);
스크립트	"movetcp spd, acc, x, y, z, rx, ry, rz"
설명	입력된 TCP 의 위치와 회전각 값을 목표로 Cartesian Space 에서 움직입니다. x, y, z 는 TCP 의 위치 값으로 mm 단위입니다.

	rx, ry, rz 는 TCP 의 방향을 결정하기 위한 Yaw-Pitch-Roll 순서의 오일러 각도를 degree 단위로 표시한 값입니다. spd 와 acc 는 각각 속도와 가속도로 0~1 로 정규화 된 값을 가집니다. -1 을 입력할 경우에는 기본 세팅 된 속도와 가속도가 이용됩니다.

함수	MoveCircle_ThreePoint(int type, float x1, float y1, float z1, float rx1, float ry1, float rz1, float x2, float y2, float z2, float rx2, float ry2, float rz2, float spd = -1, float acc = -1);
스크립트	“movecircle threepoints intended spd, acc, x1, y1, z1, rx1, ry1, rz1, x2, y2, z2, rx2, ry2, rz2” “movecircle threepoints constant spd, acc, x1, y1, z1, rx1, ry1, rz1, x2, y2, z2, rx2, ry2, rz2” “movecircle threepoints radial spd, acc, x1, y1, z1, rx1, ry1, rz1, x2, y2, z2, rx2, ry2, rz2”
설명	TCP 의 현재위치, 중간 위치, 그리고 최종 위치의 3 가지 포인트를 이용해서 원을 그리는 동작 명령입니다. type 에 따라서 TCP 의 방향을 3 가지로 결정할 수 있습니다. type=0: 스크립트에는 ‘intended’로 사용됩니다. 사용자가 입력한 방향을 지키면서 움직입니다. type=1: 스크립트에는 ‘constant’로 사용됩니다. TCP 의 현재 방향을 유지한 채로 원을 그립니다. type=2: 스크립트에는 ‘radial’로 사용됩니다. 세개의 위치로 결정되는 원의 중심방향으로 TCP 의 방향이 상대적으로 변합니다. 방향의 기준은 현재 TCP 의 방향이 시작 기준입니다. x1, y1, z1 는 TCP 의 중간 위치 값으로 mm 단위입니다. rx1, ry1, rz1 는 TCP 의 중간 위치의 방향을 결정하기 위한 Yaw-Pitch-Roll 순서의 오일러 각도를 degree 단위로 표시한 값입니다.

	x2, y2, z2 는 TCP 의 최종 위치 값으로 mm 단위입니다. rx2, ry2, rz2 는 TCP 의 최종 위치의 방향을 결정하기 위한 Yaw-Pitch-Roll 순서의 오일러 각도를 degree 단위로 표시한 값입니다. spd 와 acc 는 각각 속도와 가속도로 0~1 로 정규화 된 값을 가집니다. -1 을 입력할 경우에는 기본 세팅 된 속도와 가속도가 이용됩니다.

함수	MoveCircle_Axis(int type, float cx, float cy, float cz, float ax, float ay, float az, float rot_angle, float spd = -1, float acc = -1);
스크립트	"movecircle axis intended spd, acc, rot_angle, cx, cy, cz, ax, ay, az" "movecircle axis constant spd, acc, rot_angle, cx, cy, cz, ax, ay, az" "movecircle axis radial spd, acc, rot_angle, cx, cy, cz, ax, ay, az"
설명	TCP 의 현재위치에서 시작해서 회전 축을 중심으로 입력한 각도만큼 원을 그리는 동작 명령입니다. type 에 따라서 TCP 의 방향을 3 가지로 결정할 수 있습니다. type=0: 스크립트에는 'intended'로 사용됩니다. 사용자가 입력한 방향을 지키면서 움직입니다. type=1: 스크립트에는 'constant'로 사용됩니다. TCP 의 현재 방향을 유지한 채로 원을 그립니다. type=2: 스크립트에는 'radial'로 사용됩니다. 세개의 위치로 결정되는 원의 중심방향으로 TCP 의 방향이 상대적으로 변합니다. 방향의 기준은 현재 TCP 의 방향이 시작 기준입니다. cx, cy, cz 는 회전 중심점의 위치 값으로 mm 단위입니다. ax, ay, az 는 회전의 축을 나타내는 벡터입니다. rot_angle 은 회전할 양을 나타내는 값으로 degree 단위입니다. spd 와 acc 는 각각 속도와 가속도로 0~1 로 정규화 된 값을 가집니다. -1 을 입력할 경우에는 기본 세팅 된 속도와 가속도가 이용됩니다.

함수	MoveJointBlend_Clear(void);
스크립트	"blend_jnt clear_pt"
설명	Joint Space 에서 연속된 목적 위치들을 연결하는 동작 명령 중에서 저장된 목적 위치들을 지우는 명령어입니다.

함수	MoveJointBlend_AddPoint(float joint1, float joint2, float joint3, float joint4, float joint5, float joint6, float spd = -1, float acc = -1);
스크립트	"blend_jnt add_pt spd, acc, joint1, joint2, joint3, joint4, joint5, joint6"
설명	Joint Space 에서 연속된 목적 위치들을 연결하는 동작 명령 중에서 목적 위치들을 입력하는 명령어입니다. joint1~joint6 은 목적 관절 값이고, degree 단위입니다. spd 와 acc 는 각각 속도와 가속도로 0~1 로 정규화 된 값을 가집니다. -1 을 입력할 경우에는 기본 세팅 된 속도와 가속도가 이용됩니다. 마지막으로 입력된 spd 와 acc 로 전체 명령이 구동됩니다.

함수	MoveJointBlend_MovePoint(void);
스크립트	"blend_jnt move_pt"
설명	Joint Space 에서 연속된 목적 위치들을 연결하는 동작 명령 중에서 동작을 시작하는 명령어입니다. MoveJointBlend_AddPoint 로 입력한 위치들로 차례대로 움직이는 동작이 수행됩니다.

함수	MoveTCPBlend_Clear(void);
스크립트	“blend_tcp clear_pt”
설명	Cartesian Space 에서 연속된 목적 위치들을 연결하는 동작 명령 중에서 저장된 목적 위치들을 지우는 명령어입니다.

함수	MoveTCPBlend_AddPoint(float radius, float x, float y, float z, float rx, float ry, float rz, float spd = -1, float acc = -1);
스크립트	“blend_tcp add_pt spd, acc, radius, x, y, z, rx, ry, rz”
설명	Cartesian Space 에서 연속된 목적 위치들을 연결하는 동작 명령 중에서 목적 위치들을 입력하는 명령어입니다. radius 는 목적 위치들을 연달아 이동할 때 곡선을 그리며 다음 목적 위치로 이동하기 위한 거리를 나타낸 값으로 mm 단위입니다. 하나의 목적 위치에서 다음 목적 위치로의 이동을 선분으로 생각할 때 그 절반 값이 radius 가 가질 수 있는 최대 값입니다. 이 값보다 크게 주면 내부적으로 알아서 값을 줄여서 사용합니다. x, y, z 는 TCP 의 위치 값으로 mm 단위입니다. rx, ry, rz 는 TCP 의 방향을 결정하기 위한 Yaw-Pitch-Roll 순서의 오일러 각도를 degree 단위로 표시한 값입니다. spd 와 acc 는 각각 속도와 가속도로 0~1 로 정규화 된 값을 가집니다. -1 을 입력할 경우에는 기본 세팅 된 속도와 가속도가 이용됩니다. 마지막으로 입력된 spd 와 acc 로 전체 명령이 구동됩니다.

함수	MoveTCPBlend_MovePoint(void);
스크립트	"blend_tcp move_pt"
설명	Cartesian Space 에서 연속된 목적 위치들을 연결하는 동작 명령 중에서 동작을 시작하는 명령어입니다. MoveTCPBlend_AddPoint 로 입력한 위치들로 차례대로 움직이는 동작이 수행됩니다.

함수	ControlBoxDigitalOut(int d0, int d1, int d2, int d3, int d4, int d5, int d6, int d7, int d8, int d9, int d10, int d11, int d12, int d13, int d14, int d15)
스크립트	"digital_out d0, d1, d2, d3, d4,d5, d6, d7, d8, d9, d10, d11, d12, d13, d14, d15"
설명	컨트롤 박스의 디지털 출력 포트를 통해서 신호를 내보냅니다. d0~d15 는 각각의 디지털 포트의 on/off 를 나타냅니다. 값이 1 이면 on 이고, 0 이면 off 입니다.

함수	ControlBoxAnalogOut(float a0, float a1, float a2, float a3)
스크립트	"analog_out a0, a1, a2, a3"
설명	컨트롤 박스의 아날로그 출력 포트를 통해서 신호를 내보냅니다. a0~a3 은 각각의 아날로그 포트의 출력 값을 나타냅니다. 값은 0 에서 10 의 값을 가지며 voltage 단위입니다.

함수	ToolOut(int volt, int d0, int d1)
스크립트	"tool_out volt, d0, d1"
설명	툴 플랜지의 전압 레벨을 설정하고, 디지털 출력 포트를 통해서 신호를 내보냅니다. volt 는 툴 플랜지의 디지털 출력의 전압 레벨을 설정합니다. 값은 0, 12, 24 가 가능하고 voltage 단위입니다. 이 세개의 값 이외의 입력은 무시됩니다. d0, d1 은 각각의 디지털 포트의 on/off 를 나타냅니다. 값이 1 이면 on 이고, 0 이면 off 입니다.

함수	ProgramMode_Real(void)
스크립트	"pgmode real"
설명	로봇의 동작 상태를 리얼 모드로 변경합니다. 리얼 모드에서는 동작 명령을 내릴 때 실제로 로봇이 움직입니다.

함수	ProgramMode_Simulation(void)
스크립트	"pgmode simulation"
설명	로봇의 동작 상태를 시뮬레이션 모드로 변경합니다. 시뮬레이션 모드에서는 동작 명령을 내릴 때 로봇이 움직이지는 않습니다.

함수	BaseSpeedChange(float spd)
스크립트	“sdw default_speed spd”
설명	입력된 spd 에 모든 동작 명령의 속도가 곱해져서 동작합니다. 즉, 전체 동작의 속도를 결정하는 값입니다. spd 는 0~1 로 정규화 된 값을 가집니다.

함수	MotionPause(void)
스크립트	“task pause”
설명	로봇 동작을 일시정지 시킵니다. MotionResume 으로 동작을 재개시켜 주거나 MotionHalt 로 동작을 완전히 끝내기 전에는 다른 동작 명령을 주어도 무시됩니다.

함수	MotionResume(void)
스크립트	“task resume_a”
설명	MotionPause 로 인해 일시 정지된 로봇 동작을 재개합니다. 외부 충돌로 인해 정지된 동작을 재개시키지는 못합니다. 이 경우에는 CollisionResume 을 수행해야 합니다.

함수	CollisionResume(void)
스크립트	“task resume_b”
설명	외부 충돌로 인해 일시 정지된 로봇 동작을 재개합니다. MotionPause 로 인해 정지된 동작을 재개시키지는 못합니다. 이 경우에는 MotionResume 을 수행해야 합니다.

함수	MotionHalt(void)
스크립트	“task stop”
설명	로봇의 동작을 완전히 멈춥니다. 또한 MotionPause 혹은 외부 충돌로 인해 일시 정지된 로봇의 동작을 완전히 멈춥니다.

```
int test_flag = false;
int test_state = 0;
void MainWindow::onLogic(){

    // Motion Sequence ===============================================
    if(test_flag == true){
        switch(test_state){
        case 0:
            if(IsMotionIdle()){
                print("test_state 0\n");
                MoveJoint(0,0,0,0,0,0);
                test_state = 1;
            }
            break;

        case 1:
            if(IsMotionIdle()){
                print("test_state 1\n");
                MoveJoint(50,50,50,50,50,50);
                test_state = 2;
            }
            break;
        case 2:
            if(IsMotionIdle()){
                print("test_state 2\n");
                MoveTCP(400,400,400,150,30,-100);
                test_state = 3;
            }
            break;
        case 3:
            if(IsMotionIdle()){
                print("test_state 3\n");
                test_flag = false;
                test_state = 0;
            }
            break;
        }
    }
    // ===============================================================

}
```

위의 코드는 두개의 관절 제어 모션과 하나의 TCP 제어 모션을 순차적으로 수행하는 동작 코드입니다. 동작 시퀀스를 수행할지 말지 결정하는 'test_flag'가 있고, 만약 이 값이 참일 경우에 시퀀스의 진행을 나타내는 'test_state'값에 따라서 이전 모션에서 다음 모션으로 순차적으로 넘어 갑니다.

이때 이전 모션이 끝났는지 끝나지 않았는지 확인을 해야 하고, 이를 쉽게 하기 위해 제공하는 'IsMotionIdle' 함수가 있습니다. 이 함수는 명령어를 로봇의 메인 제어기로 보내고 응답 메시지의 수신 여부와 현재 로봇 상태가 'Idle'인지 'Moving'인지 여부를 확인합니다.

```
int MainWindow::IsMotionIdle(){
   return ((cmdConfirmFlag == true) && (systemStat.sdata.robot_state == 1));
}
```

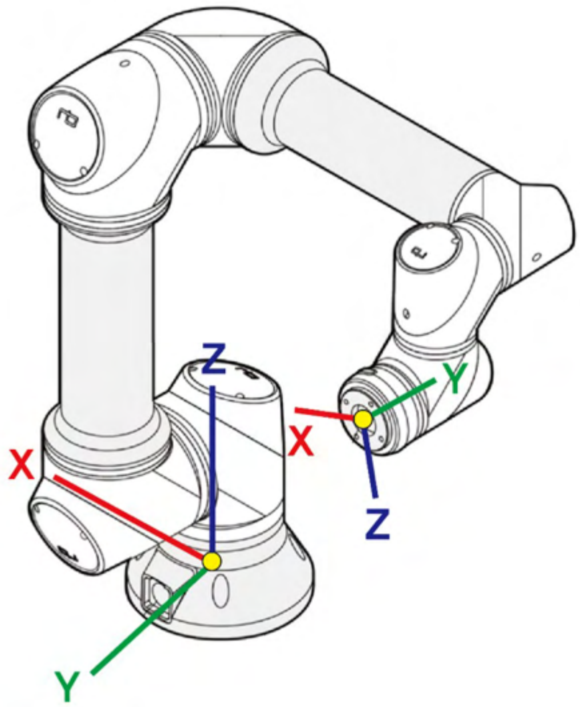
동작 시퀀스가 들어있는 'onLogic' 함수는 Qt 에서 제공하는 타이머에 연동되어 있는 함수로, 예제에서는 10ms 주기로 설정되어, 매 10ms 마다 이 함수가 실행됩니다.

```
void MainWindow::on_BTN_TEST_clicked()
{
    test_state = 0;
    test_flag = true;
}
```

로봇 동작을 실행시키는 부분은 간단합니다. 동작 시퀀스 상태를 나타내는 'test_state' 값을 동작의 시작 지점인 0 으로 세팅하고, 동작을 수행하는 'test_flag' 값을 true 로 주면 로봇이 움직입니다.

제공된 예제 코드에서 보여주는 동작은 매우 단순하지만 구조에 얽매임이 없이 자유롭습니다. 사용자는 이 예제 코드를 기반으로 원하는 어플리케이션을 구현할 수도 있고, 스크립트만 이해해서 별도의 어플리케이션을 구현할 수도 있습니다.

부록 F 좌표계 설정



- **Global 좌표계**

  로봇의 베이스에 고정된 좌표계로 로봇이 한번 고정되면 Global 좌표계 또한 고정됩니다.

  베이스 기저면의 중심을 원점으로 한다. 원점에서 로봇 방향을 +Z 방향, 원점에서 커넥터 방향을 +Y 방향으로 정합니다.

- **Local 좌표계**

  로봇의 TCP(Tool Center Point)에 고정된 좌표계로서 TCP 의 오프셋을 설정하거나 이동함에 따라 실시간으로 축의 방향이 바뀝니다.

  TCP 를 원점으로 하며, 원점에서 로봇방향을 +Y 방향, 원점에서 직접 교시 버튼 방향을 +Z 방향으로 정합니다.

부록 G 정지 시간 및 정지 거리

RB Series 협동 로봇에서는 로봇 정지시간 및 정지 거리가 안전 모니터링 기능에 의해 생성됩니다.

아래 그래프는 Joint 0(Base 축), Joint 1(Shoulder 축), Joint 2(Elbow 축)에 대한 정지 카테고리 0 의 정지 시간과 정지 거리를 나타냅니다.

주의

상황에 따라 실제 정지 움직임은 아래의 결과와는 상이할 수 있습니다. Joint 0 의 경우 수평 움직임에 대해 측정된 결과이며, Joint 1 과 2 의 경우는 수직 하강 움직임 상황에서 측정된 결과입니다. 팔의 길이의 경우 최대 거리가 적용되었습니다.

RB5-850E Series Base (Joint 0)		
	Stop Distance (mm)	Stop Time (sec)
Test 1	183.38	0.19
Test 2	160.1	0.24
Test 3	191.03	0.24
Maximum	191.03	0.24
Average	178.17	0.22
Condition	Max. Reach / Max. Velocity / Horizontal Motion	

RB5-850E Series Shoulder (Joint 1)		
	Stop Distance (mm)	Stop Time (sec)
Test 1	183.71	0.13
Test 2	177.53	0.13
Test 3	183.39	0.21
Maximum	183.71	0.21
Average	181.54	0.16
Condition	Max. Reach / Max. Velocity / Vertical Downward Motion	

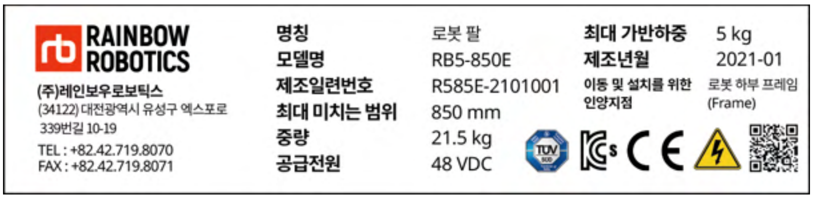
RB5-850E Series Elbow (Joint 2)		
	Stop Distance (mm)	Stop Time (sec)
Test 1	90.935	0.14
Test 2	94.809	0.14
Test 3	81.987	0.14
Maximum	94.809	0.14
Average	89.24	0.14
Condition	Max. Reach / Max. Velocity / Vertical Downward Motion	

부록 H 명판 표기

로봇의 명판은 다음과 같이 로봇 팔과 컨트롤 박스로 구분하여 표기합니다.

[로봇 팔]

RB5-850E Series

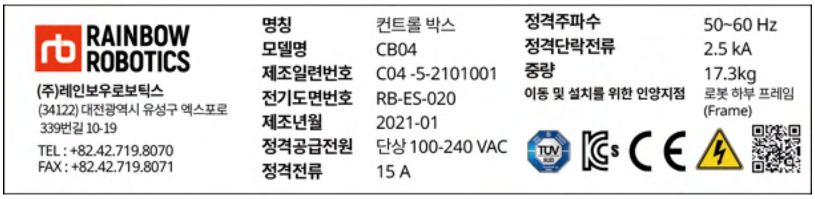


RB3-1200E Series

RB10-1300E Series

[컨트롤 박스]

RB5-850E Series, RB3-1200E Series: Stand type(CB04)



부록 I 모드 버스 서버

- **주의**

  본 매뉴얼은 모드버스 서버(slave)에 관한 설명입니다. 모드버스 클라이언트 기능에 관한 설명은 6 단원을 참고해 주십시오.

1. 개요

RB Series 의 모드버스 TCP 서버(slave)는 포트번호 502 로 고정되어 있습니다. IP 주소는 UI 를 통한 네트워크 설정에 따라 변경됩니다(초기 IP 주소: 10.0.2.7).

RB Series 의 모드버스 서버는 다중 client 의 접속을 허용하고, 아래와 같은 동작 명령에 대해 수행합니다.

	Function Code	Function Name
Bit Address	2	Read Discrete Inputs
	1	Read Coils
	5	Write Single Coil
	15	Write Multiple Coils
16-bit (Word) Address	4	Read Input Registers
	3	Read Multiple Holding Registers
	6	Write Single Holding Register
	16	Write Multiple Holding Registers

2. Exception Code

잘못된 주소에 대해 접근하거나 값의 범위가 잘못되었거나 혹은 잘못된 명령어를 보냈을 경우에 아래와 같은 에러 메시지를 반환합니다.

Exception Code	Exception Name
1	Illegal Function
2	Illegal Data Address
3	Illegal Value

3. Bit Address Map

Bit Address			
Address	Function	Read	Write
0	Box digital input 0	o	x
1	Box digital input 1	o	x
2	Box digital input 2	o	x
3	Box digital input 3	o	x
4	Box digital input 4	o	x
5	Box digital input 5	o	x
6	Box digital input 6	o	x
7	Box digital input 7	o	x
8	Box digital input 8	o	x
9	Box digital input 9	o	x
10	Box digital input 10	o	x
11	Box digital input 11	o	x
12	Box digital input 12	o	x
13	Box digital input 13	o	x
14	Box digital input 14	o	x
15	Box digital input 15	o	x
16	Box digital output 0	o	o
17	Box digital output 1	o	o
18	Box digital output 2	o	o
19	Box digital output 3	o	o
20	Box digital output 4	o	o
21	Box digital output 5	o	o
22	Box digital output 6	o	o
23	Box digital output 7	o	o
24	Box digital output 8	o	o
25	Box digital output 9	o	o
26	Box digital output 10	o	o
27	Box digital output 11	o	o
28	Box digital output 12	o	o
29	Box digital output 13	o	o
30	Box digital output 14	o	o
31	Box digital output 15	o	o
32	Tool digital input 0	o	x
33	Tool digital input 1	o	x
34	Tool digital output 0	o	o
35	Tool digital output 1	o	o

4. Word(16 bit) Address Map

Word Address				
Address	Function	Read	Write	Comments
0	Box digital input 0~15	o	x	[BBBB BBBB BBBB BBBB]
1	Box digital output 0~15	o	o	[BBBB BBBB BBBB BBBB]
2	Box analog input 0	o	x	1mV unit
3	Box analog input 1	o	x	1mV unit
4	Box analog input 2	o	x	1mV unit
5	Box analog input 3	o	x	1mV unit
6	Box analog output 0	o	o	1mV unit
7	Box analog output 1	o	o	1mV unit
8	Box analog output 2	o	o	1mV unit
9	Box analog output 3	o	o	1mV unit
10	Extend digital input 0~15	o	x	[BBBB BBBB BBBB BBBB]
11	Extend digital output 0~15	o	x	[BBBB BBBB BBBB BBBB]
12	Extend analog input 0	o	x	1mV unit
13	Extend analog input 1	o	x	1mV unit
14	Extend analog input 2	o	x	1mV unit
15	Extend analog input 3	o	x	1mV unit
16	Extend analog output 0	o	o	1mV unit
17	Extend analog output 1	o	o	1mV unit
18	Extend analog output 2	o	o	1mV unit
19	Extend analog output 3	o	o	1mV unit
20~29	Reserved (Box IO)			
30	Tool output voltage	o	o	0, 12, 24
31	Tool digital input 0~1	o	x	[TTxx xxxx xxxx xxxx]
32	Tool digital output 0~1	o	o	[TTxx xxxx xxxx xxxx]
33	Tool analog input 0	o	x	1mV unit
34	Tool analog input 1	o	x	1mV unit
35~49	Reserved (Tool IO)			
50	Is Robot Activated	o	x	0 or 1
51	Is Real-mode	o	x	0 or 1
52	Is Collision Detected	o	x	0 or 1
53	Is Robot arm power engaged	o	x	0 or 1

54	Is Direct Teaching mode	o	x	0 or 1
55	Is Robot moving	o	x	0 or 1
56	Is Pause state	o	x	0 or 1
57	Is Teaching pendant is connected	o	x	0 or 1
58	Is Program Run	o	x	0 or 1
59	Is No-Arc mode is on	o	x	0 or 1
60	Is EMG button released	o	x	0 or 1
61	Is First Program Run	o	x	0 or 1
62~99	Reserved (Future System)			
100	Command: Start Program Once	o	o	Rising Edge is command
101	Command: Start Program Repeat	o	o	Rising Edge is command
102	Command: Pause Program	o	o	Rising Edge is command
103	Command: Stop Program	o	o	Rising Edge is command
104	Command: Resume from pause	o	o	Rising Edge is command
105	Command: Resume from collision	o	o	Rising Edge is command
106	Command: Load default Program	o	o	Rising Edge is command
107	Command: Robot Arm activation	o	o	Rising Edge is command
108	Command: Change to Real-mode	o	o	Rising Edge is command
109	Command: Power off the robot arm	o	o	Rising Edge is command
110~127	Reserved (Future System)			
128~255	User General Purpose Register	o	o	User Define Area
256	Joint reference 0	o	x	0.02deg unit / Signed
257	Joint reference 1	o	x	0.02deg unit / Signed
258	Joint reference 2	o	x	0.02deg unit / Signed
259	Joint reference 3	o	x	0.02deg unit / Signed
260	Joint reference 4	o	x	0.02deg unit / Signed
261	Joint reference 5	o	x	0.02deg unit / Signed
262	Joint angle 0	o	x	0.02deg unit / Signed
263	Joint angle 1	o	x	0.02deg unit / Signed
264	Joint angle 2	o	x	0.02deg unit / Signed
265	Joint angle 3	o	x	0.02deg unit / Signed
266	Joint angle 4	o	x	0.02deg unit / Signed
267	Joint angle 5	o	x	0.02deg unit / Signed

268	Joint current 0	o	x	10mA unit / Signed
269	Joint current 1	o	x	10mA unit / Signed
270	Joint current 2	o	x	10mA unit / Signed
271	Joint current 3	o	x	10mA unit / Signed
272	Joint current 4	o	x	10mA unit / Signed
273	Joint current 5	o	x	10mA unit / Signed
274	Joint information 0	o	x	
275	Joint information 1	o	x	
276	Joint information 2	o	x	
277	Joint information 3	o	x	
278	Joint information 4	o	x	
279	Joint information 5	o	x	
280	Joint temperature 0	o	x	celcius unit
281	Joint temperature 1	o	x	celcius unit
282	Joint temperature 2	o	x	celcius unit
283	Joint temperature 3	o	x	celcius unit
284	Joint temperature 4	o	x	celcius unit
285	Joint temperature 5	o	x	celcius unit
286	Joint 0 Estimated Current	o	x	10mA unit / Signed
287	Joint 1 Estimated Current	o	x	10mA unit / Signed
288	Joint 2 Estimated Current	o	x	10mA unit / Signed
289	Joint 3 Estimated Current	o	x	10mA unit / Signed
290	Joint 4 Estimated Current	o	x	10mA unit / Signed
291	Joint 5 Estimated Current	o	x	10mA unit / Signed
292	Joint 0 Gap(Esti.-Meas.) Current	o	x	10mA unit / Signed
293	Joint 1 Gap(Esti.-Meas.) Current	o	x	10mA unit / Signed
294	Joint 2 Gap(Esti.-Meas.) Current	o	x	10mA unit / Signed
295	Joint 3 Gap(Esti.-Meas.) Current	o	x	10mA unit / Signed
296	Joint 4 Gap(Esti.-Meas.) Current	o	x	10mA unit / Signed
297	Joint 5 Gap(Esti.-Meas.) Current	o	x	10mA unit / Signed
298	Joint 0 Gap(Esti.-Meas.) Curr+LPF	o	x	10mA unit / Signed
299	Joint 1 Gap(Esti.-Meas.) Curr+LPF	o	x	10mA unit / Signed

300	Joint 2 Gap(Esti.-Meas.) Curr+LPF	o	x	10mA unit / Signed
301	Joint 3 Gap(Esti.-Meas.) Curr+LPF	o	x	10mA unit / Signed
302	Joint 4 Gap(Esti.-Meas.) Curr+LPF	o	x	10mA unit / Signed
303	Joint 5 Gap(Esti.-Meas.) Curr+LPF	o	x	10mA unit / Signed
304~329	Reserved (Joint Information)			
330	TCP reference X	o	x	0.1mm unit / Signed
331	TCP reference Y	o	x	0.1mm unit / Signed
332	TCP reference Z	o	x	0.1mm unit / Signed
333	TCP reference RX	o	x	0.02deg unit / Signed
334	TCP reference RY	o	x	0.02deg unit / Signed
335	TCP reference RZ	o	x	0.02deg unit / Signed
336	TCP position X	o	x	0.1mm unit / Signed
337	TCP position Y	o	x	0.1mm unit / Signed
338	TCP position Z	o	x	0.1mm unit / Signed
339	TCP position RX	o	x	0.02deg unit / Signed
340	TCP position RY	o	x	0.02deg unit / Signed
341	TCP position RZ	o	x	0.02deg unit / Signed
342~389	Reserved (TCP Information)			

부록 J 시스템 업데이트

- **주의**

  시스템 업데이트 이전에 태블릿 UI 내부의 프로그램 파일 (.wsl)을 백업해 두는 것을 권장합니다.

1. 개요

시스템 업데이트는 크게 두 단계로 이루어 집니다.

APK 설치를 통한 UI 프로그램 업데이트 → 시스템 소프트웨어 업데이트

2. 작업 프로그램 백업

태블릿과 개인/업무용 PC 를 연결하여 아래의 경로에서 프로그램 파일 (.wsl)을 획득 후 백업합니다.

Tablet → Android → data → com.rainbow.cobot → files → work→ 작업할 wsl 획득

※ 다음 단계 진행전 획득한 파일을 백업해 두는 것을 권장합니다.

3. UI 프로그램 업데이트

태블릿 UI/메인 시스템 프로그램은 APK 형태로 배포됩니다.

이는 일반 안드로이드 어플리케이션과 동일한 형태의 설치 파일입니다. 따라서 설치 APK 파일을 태블릿으로 이동 후 설치하면 UI 프로그램이 업데이트 됩니다.

※ 레인보우로보틱스에서는 기존의 어플리케이션을 삭제 후 설치를 권장함

※ 기존의 어플리케이션 삭제 시 프로그램 파일(.wsl)이 같이 삭제되기 때문에, 1 과정에서 프로그램 파일을 백업 후 본 과정을 진행하기 바람

배포된 APK 파일을 태블릿으로 이동 → APK 설치

4. 태블릿과 컨트롤 박스 연결

태블릿을 컨트롤 박스에 연결하고 UI 프로그램에 접속합니다. 접속 후 태블릿과 컨트롤 박스 통신을 연결합니다.

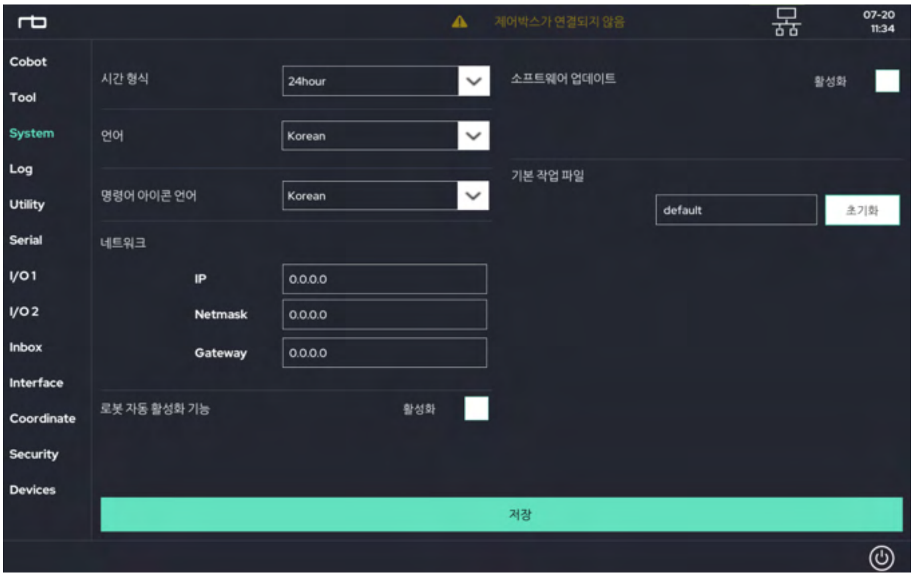
UI 홈 화면 → Make → 좌측 하단 'State'클릭 → Connect

태블릿과 컨트롤 박스 사이 통신이 정상 연결되면 첫 번째 칸에 파란 불이 들어옵니다. 안전상, 로봇 초기화는 진행하지 않는 것을 권장합니다.

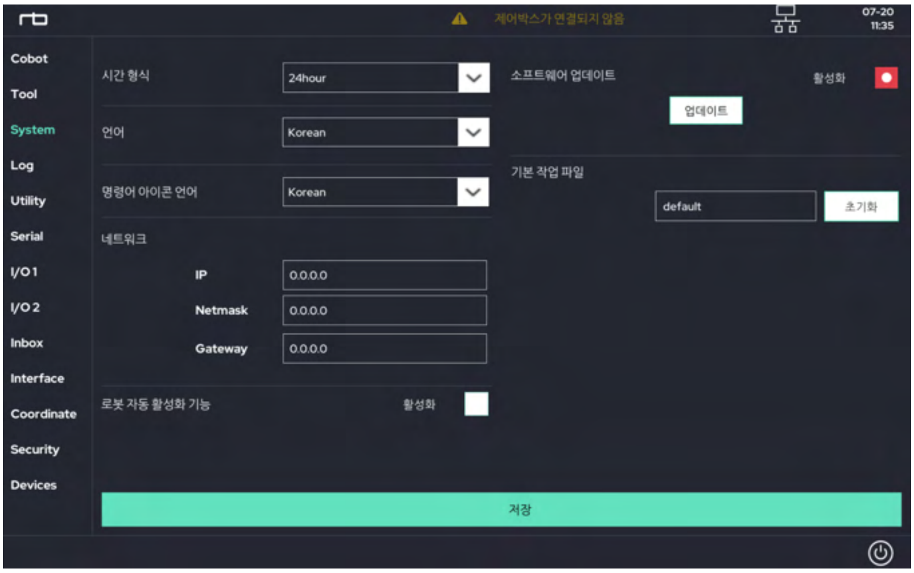
5. 시스템 소프트웨어 업데이트 영역으로 이동 및 활성화

아래와 같이 시스템 소프트웨어 경로로 이동합니다.

UI 홈 화면 → Make → 하단의 Page 선택 → Setup → System 탭으로 이동

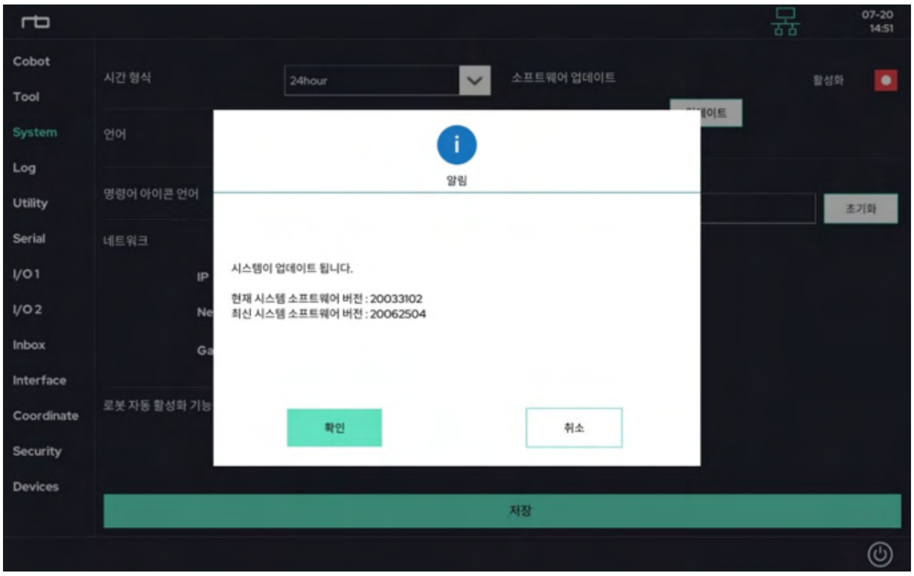


우측에 있는 “Software Update” 구역에서 Activate(활성화) 체크박스를 클릭합니다.



6. 시스템 소프트웨어 업데이트 진행

“Update”버튼이 나타나고, 이 버튼을 클릭하면 팝업 창이 뜹니다.



“OK” 버튼을 누르고 기다리면 소프트웨어가 업데이트 됩니다.

OK 버튼 클릭 이후, 정상적으로 업데이트가 완료되면, 5~15 초 이내에 컨트롤 박스 (제어기)의 PC (프로그램)이 자동으로 재시작 됩니다.

재시작 과정에서 컨트롤 박스의 LCD 창에 “Please Wait…”가 일시적으로 표시됩니다. 이는 컨트롤 박스가 다시 부팅 중임을 알려줍니다.

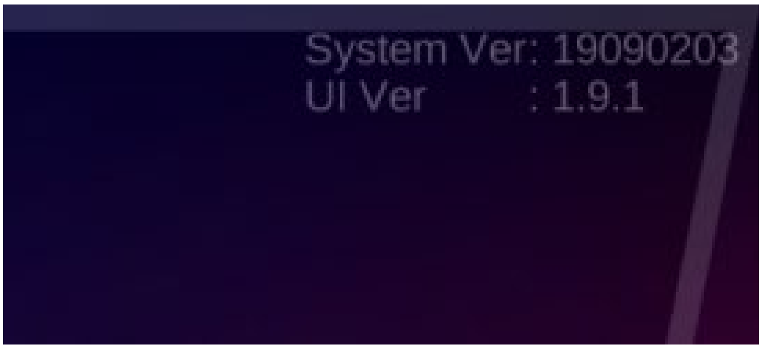
이후, 최종적으로 재부팅이 완료되면 컨트롤 박스의 LCD 에 “Normal Operation”이 표시됩니다.

7. 소프트웨어 업데이트 확인

UI 태블릿과 컨트롤 박스를 다시 연결합니다.

UI 홈 화면 → Make → 좌측 하단 ‘State’클릭 → Connect

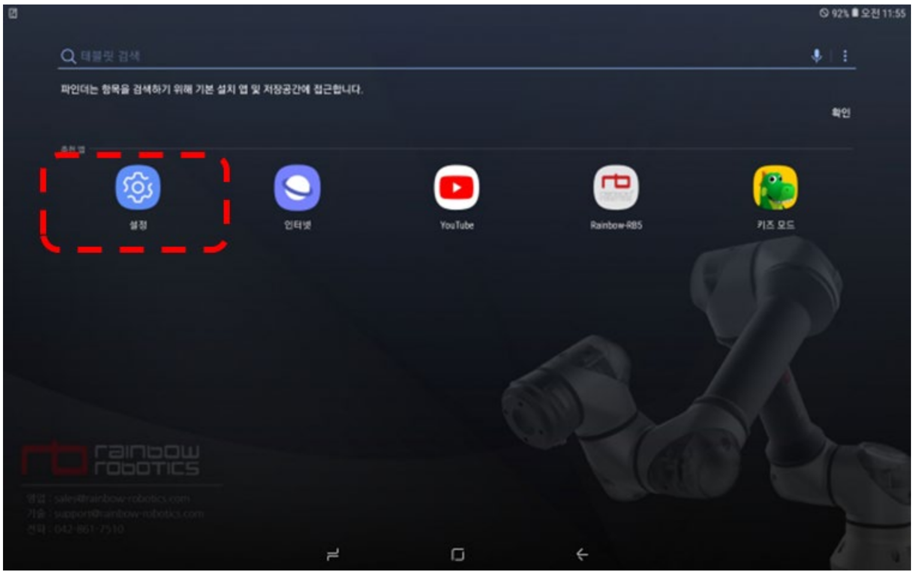
UI 의 홈 화면으로 다시 나가보면 우측 상단(또는 좌측 하단)에 소프트웨어 버전이 표기되어 있습니다. 알맞은 버전으로 업데이트 되었는지 확인합니다.



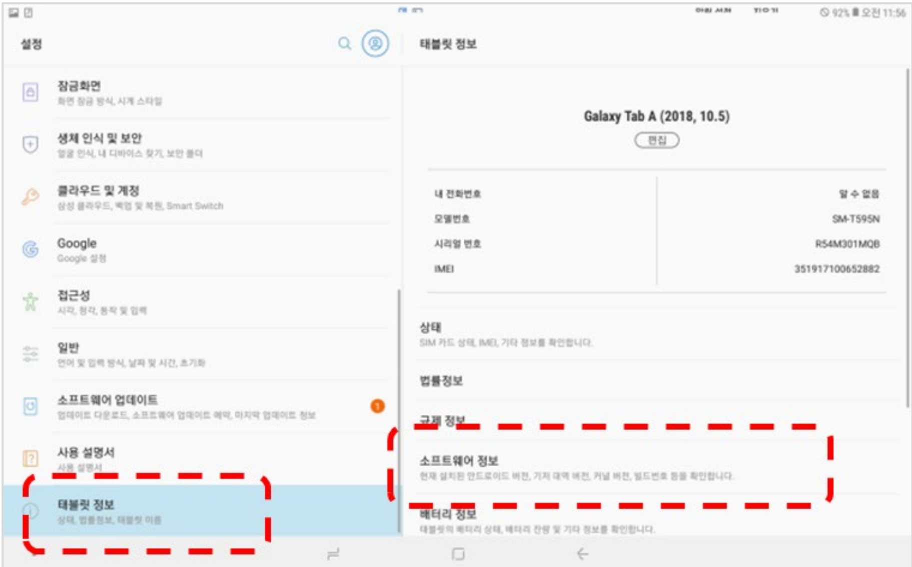
부록 K 안드로이드 태블릿 설정

UI 프로그램 사용 이전에, 다음과 같은 태블릿 설정이 요구됩니다.

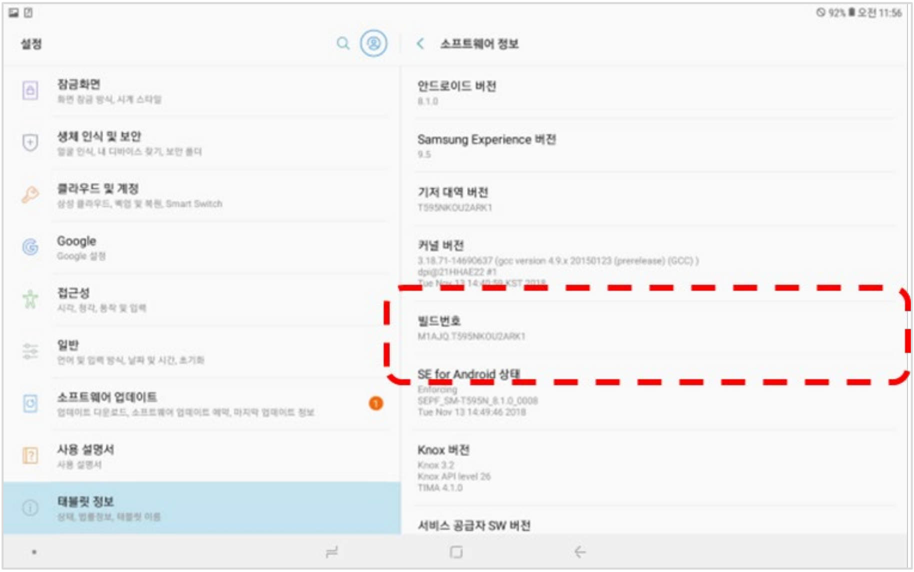
1. **안드로이드 "설정"으로 이동합니다.**



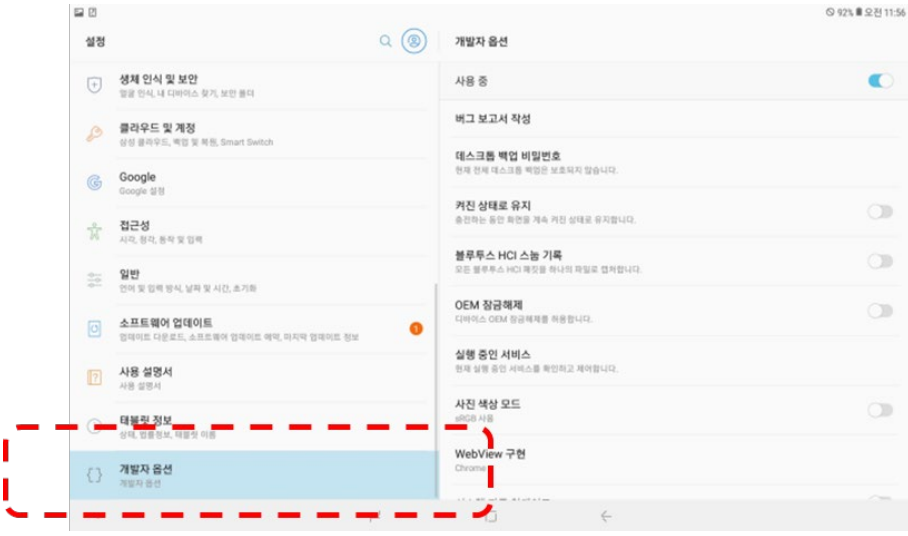
2. 설정 메뉴 중 "태블릿 정보" → "소프트웨어 정보"를 클릭합니다.



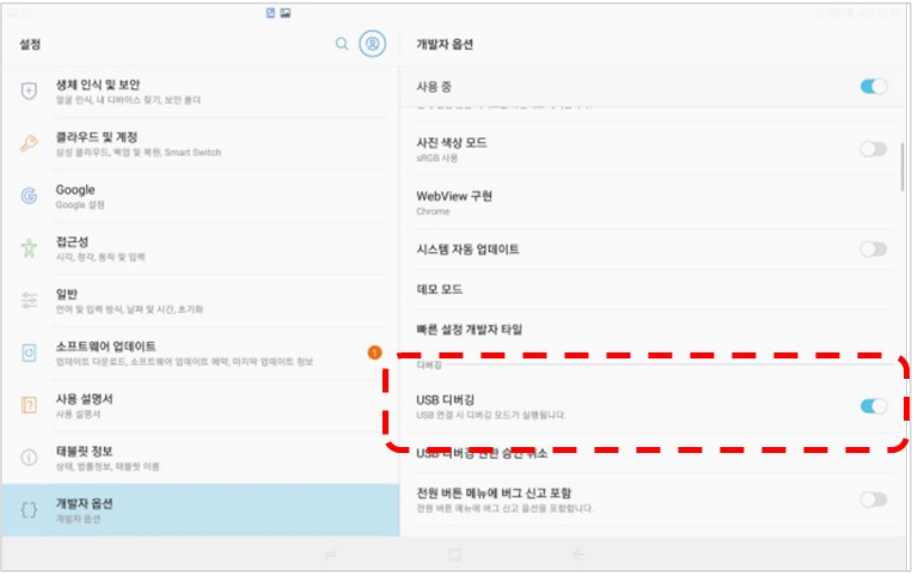
4. 태블릿 정보 중 “빌드 번호” 다중 클릭(7 회 이상)합니다.



5. 아래와 같이 “태블릿 정보” 아래에 “개발자 옵션” 이라는 메뉴가 등장합니다.



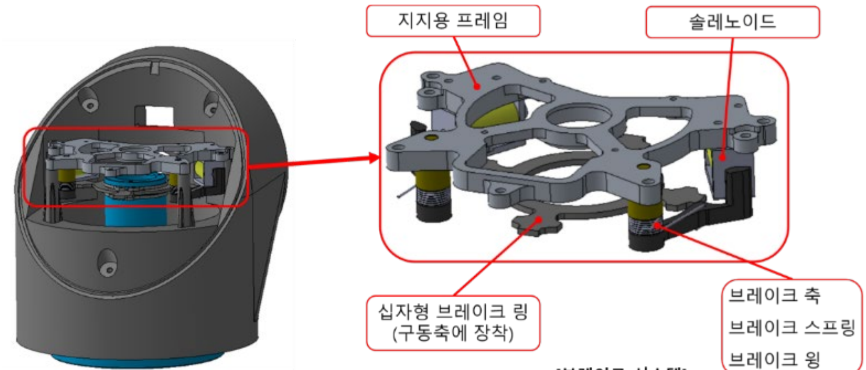
6. “개발자 옵션” 중, “USB 디버깅”을 활성화시켜줍니다.



7. 레인보우로보틱스에서 배포한 APK 를 실행하여 UI 프로그램을 태블릿에 설치합니다.

부록 L 기계적 멈춤 장치(BRAKE SYSTEM)

로봇 팔의 각 축에 설치된 기계적 멈춤 장치(Brake System)의 구성은 지지용 프레임, 솔레노이드, 브레이크 링, 브레이크 축, 브레이크 스프링, 브레이크 윙으로 구성되어 있으며, 로봇 관절에 아래의 그림과 같이 설치되어 있다.



[로봇 관절] [브레이크 시스템]

솔레노이드가 ON 되는 경우에 브레이크 링의 회전반경과 브레이크 윙 사이의 물리적 간섭이 해제되고, 솔레노이드가 OFF 되는 경우에는 브레이크 링의 말단부와 브레이크 윙 사이의 물리적 간섭이 발생하여 구동축의 회전이 정지하게 된다.

브레이크 링이 회전하며 브레이크 윙을 밀고 지나가면 윙은 스프링의 힘으로 복귀하며, 이후 물리적 간섭을 통해 양방향 브레이크가 발생하여 구동축의 양방향 회전이 모두 정지된 상태로 유지된다.

부록 M 산업안전보건기준에 관한 규칙 및 안전검사고시

– 산업안전보건기준에 관한 규칙 제 13 절 산업용 로봇

제222조(교시 등) 사업주는 산업용 로봇(이하 "로봇"이라 한다)의 작동범위에서 해당 로봇에 대하여 교시(教示) 등[매니퓰레이터(manipulator)의 작동순서, 위치・속도의 설정・변경 또는 그 결과를 확인하는 것을 말한다. 이하 같다]의 작업을 하는 경우에는 해당 로봇의 예기치 못한 작동 또는 오(誤)조작에 의한 위험을 방지하기 위하여 다음 각 호의 조치를 하여야 한다. 다만, 로봇의 구동원을
차단하고 작업을 하는 경우에는 제2호와 제3호의 조치를 하지 아니할 수 있다.

1. 다음 각 목의 사항에 관한 지침을 정하고 그 지침에 따라 작업을 시킬 것
   가. 로봇의 조작방법 및 순서
   나. 작업 중의 매니퓰레이터의 속도
   다. 2명 이상의 근로자에게 작업을 시킬 경우의 신호방법
   라. 이상을 발견한 경우의 조치
   마. 이상을 발견하여 로봇의 운전을 정지시킨 후 이를 재가동시킬 경우의 조치
   바. 그 밖에 로봇의 예기치 못한 작동 또는 오조작에 의한 위험을 방지하기 위하여 필요한 조치

2. 작업에 종사하고 있는 근로자 또는 그 근로자를 감시하는 사람은 이상을 발견하면 즉시 로봇의 운전을 정지시키기 위한 조치를 할 것

3. 작업을 하고 있는 동안 로봇의 기동스위치 등에 작업 중이라는 표시를 하는 등 작업에 종사하고 있는 근로자가 아닌 사람이 그 스위치 등을 조작할 수 없도록 필요한 조치를 할 것

제223조(운전 중 위험 방지) 사업주는 로봇의 운전(제222조에 따른 교시 등을 위한 로봇의 운전과 제224조 단서에 따른 로봇의 운전은 제외한다)으로 인하여 근로자에게 발생할 수 있는 부상 등의 위험을 방지하기 위하여 높이 1.8미터 이상의 울타리(로봇의 가동범위 등을 고려하여 높이로 인한 위험성이 없는 경우에는 높이를 그 이하로 조절할 수 있다)를 설치하여야 하며, 컨베이어 시스템의
설치 등으로 울타리를 설치할 수 없는 일부 구간에 대해서는 안전매트 또는 광전자식 방호장치 등 감응형(感應形) 방호장치를 설치하여야 한다. 다만, 고용노동부장관이 해당 로봇의 안전기준이
「산업표준화법」 제12조에 따른 한국산업표준에서 정하고 있는 안전기준 또는 국제적으로 통용되는 안전기준에 부합한다고 인정하는 경우에는 본문에 따른 조치를 하지 아니할 수 있다.
제224조(수리 등 작업 시의 조치 등) 사업주는 로봇의 작동범위에서 해당 로봇의 수리・검사・조정 (교시 등에 해당하는 것은 제외한다)・청소・급유 또는 결과에 대한 확인작업을 하는 경우에는 해당 로봇의 운전을 정지함과 동시에 그 작업을 하고 있는 동안 로봇의 기동스위치를 열쇠로 잠근 후 열쇠를 별도 관리하거나 해당 로봇의 기동스위치에 작업 중이란 내용의 표지판을 부착하는 등 해당 작업에 종사하고 있는 근로자가 아닌 사람이 해당 기동스위치를 조작할 수 없도록 필요한 조치를 하여야 한다. 다만, 로봇의

운전 중에 작업을 하지 아니하면 안되는 경우로서 해당 로봇의 예기치 못한 작동 또는 오조작에 의한 위험을 방지하기 위하여 제222조 각 호의 조치를 한 경우에는 그러하지 아니하다.

– 안전검사고시 [별표 14] 관한 규칙 산업용로봇 검사기준

자동 운전모드

가. 자동운전모드에서는 방책 등 안전장치가 정상기능을 유지하고 있어야 하며, 보호영역 안으로 사람이 들어가면 보호정지 기능이 작동될 것

나. 자동운전모드 선택으로 보호정지 또는 비상정지가 리셋 또는 무효화 되지 않을 것

다. 정지신호가 부여되면 자동운전모드가 해제될 것

라. 자동모드를 선택하는 경우 자동모드의 기동은 보호영역 외부에서 별도의 기동조작에 의해서만 가능할 것

마. 자동운전모드에서 다른 운전모드로의 변환은 구동부가 정지된 상태에서만 가능할 것

펜던트 제어

펜던트에 줄이 달린 경우, 교시작업자가 안전하게 작업을 수행할 수 있도록 줄의 길이가 충분할 것

전기 접속 기구

전기 접속구 등 로봇에 연결되는 전기 접속장치는 임의로 분리되지 않는 방식일 것

로봇 시스템 배치설계

작업영역, 접근 및 여유 공간을 위한 로봇 시스템의 배치는 다음 각 목에 적합할 것

가. 로봇의 최대 영역을 확인하여 제한 영역 및 작업 영역을 설정하고, 로봇과 건물 기둥 등의 장애물 사이에 여유 공간이 있을 것

나. 보행자 통로 등 안전한 통행을 위한 통로가 확보되어 있을 것

다. 제어시스템 접근 및 경로가 안전할 것

라. 점검, 청소, 수리, 유지보수 등을 위한 접근 시의 안전통로가 확보되어 있을 것

마. 배선 또는 기타 위험원으로 인한 미끄러짐, 헛디딤, 넘어짐 위험이 없을 것

바. 전선 선반(cable tray)등으로 인한 위험이 없을 것

사. 자동운전 동안 접근이 필요한 운전 제어기와 보조장비(용접 제어기, 공압 밸브 등)는 보호영역 외부에 위치할 것

로봇 시스템 정지 기능

모든 로봇 시스템은 보호정지 및 별도의 비상정지 기능을 가질 것

가. 비상정지 기능

1) 로봇 시스템은 시스템의 모든 관련부분에 대하여 작동되는 단일 비상정지 기능을 가질 것

2) 다중 로봇 또는 다중 셀의 경우 제어범위를 나눌 수 있으며, 제어범위는 비상정지 장치 근처에 문자 또는 기호로 명확하게 표시되어 있을 것
3) 제어위치가 2개 이상인 경우, 각 제어위치에 설치된 비상정지장치는 항상 그 기능을 유지할 것
4) 다른 모든 로봇 제어보다 우선권을 가지며, 비상정지 시 로봇 구동기로부터 구동동력을 제거하고, 초기화되기 전까지 정지상태를 유지할 것
5) 초기화는 수동으로만 이루어져야 하며, 초기화 후에 바로 재기동되는 것이 아니라 별도의 기동조작에 의해 기동될 것
6) 비상정지회로의 성능은 제4호 안전관련 제어시스템 성능요건을 만족할 것

나. 보호정지 기능
1) 로봇 시스템은 외부 보호장치와 연결하기 위한 하나 이상의 보호정지회로를 구비하고 있을 것
2) 보호정지회로는 작동 시 로봇동작 정지, 액츄에이터 동력 제거 및 로봇에 의해 제어되는 모든 위험원에 대한 제어가 가능할 것
3) 보호정지는 수동 또는 제어로직에 의해 기동될 것
4) 적어도 하나 이상의 보호정지 기능은 0정지방식 또는 1정지방식일 것
주1) 0정지방식: 액추에어터 전원의 즉각적인 차단에 의한 정지
주2) 1정지방식: 액추에이터에는 전원이 공급된 상태에서 기계가 정지한 후 전원이 차단되는 제어정지방식
5) 보호정지회로의 성능은 제4호 안전관련 제어시스템 성능요건을 만족할 것

수동 리셋, 기동과 재기동

로봇 시스템은 예기치 않은 기동을 방지하기 위하여 다음 각 목에 적합할 것

가. 기동과 재기동은 모든 안전기능 및 보호 대책이 정상 작동 시에만 가능할 것

나. 동력 공급이 중단되었다가 재개되는 경우 위험한 동작이 자동으로 시작되는 것을 방지하기 위하여 기동 연동 장치가 설치되어야 하며, 수동조작에 의해서만 리셋될 것

다. 기동과 재기동 제어는 보호영역 밖에서 수동으로 조작되어야 하며, 보호영역 안에서는 활성화가 불가능할 것

라. 기동과 재기동 제어장치의 위치는 보호 영역을 확실하게 볼 수 있는 곳에 위치할 것. 다만, 보호 영역에 대한 시야 확보가 어려운 경우에는 다음 중 어느 하나에 따를 것
1) 보호영역 안에 있는 작업자를 검출할 수 있는 감지장치 설치
2) 출입문을 열어 놓은 상태로 고정할 수 있는 수단 설치
3) 보호영역 내부에 추가적인 시간제한 리셋 장치 설치
4) 보호영역 내부에서 충분히 인지할 수 있는 청각・시각 경고신호 발생 및 충분한 탈출시간 제공

마. 라목의 4)에 따르는 경우에는 보호영역 안에 작업자가 쉽게 접근 가능한 위치에 충분한 수의 비상정지 장치를 설치할 것

보호영역 및 방책 등

보호영역 및 방책은 다음 각 목에 적합할 것. 다만, 협동로봇은 운전 방식에 따라 일부 적용을 제외할 수 있다.보호영역 및 방책은 다음 각 목에 적합할 것. 다만, 협동로봇은 운전 방식에 따라 일부 적용을 제외할 수 있다.

가. 제한영역은 보호영역 내에 위치 할 것

나. 보호영역은 방책에 의하여 설정될 것. 다만, 재료나 부품의 투입 또는 인출을 위한 개구부의 경우에는 감응형 방호장치 등에 의하여 설정될 것

다. 로봇 셀의 방책은 다음 각목에 적합할 것

1) 방책은 외력에 의해 쉽게 파손되지 않도록 견고하게 설치하고, 도구를 사용해서만 제거할 수 있는 구조일 것
2) 방책에 재료나 부품의 투입 또는 인출을 위한 개구부가 있을 경우, 개구부의 아래, 옆 또는 개구부를 통한 작업자가 위험점에 접근하는 것을 물리적으로 방지하고, 이러한 조치가 불가능할 경우에는 감응형 방호장치 등 부가적인 조치를 할 것
3) 방책의 높이는 1,800mm 이상일 것. 다만, 로봇의 가동범위 및 KS B ISO 13857에 따른 안전거리를 고려하여 높이로 인한 위험성이 없는 경우에는 방책의 높이를 1,400mm 이상으로 할 수 있다.
4) 가동식 방책(출입문)은 옆으로 열리거나 위험원으로부터 멀어지는 방향으로 열려야 하며, 보호영역쪽으로 열리지 않는 구조일 것
5) 가동식 방책(출입문)에는 연동장치가 설치되어야 하며, 연동장치는 작업자가 위험원에 접근하기 전에 위험원을 안전한 상태로 만들 수 있을 것. 다만, 위험원이 제거되기 전에 위험지역에 접근할 가능성이 있는 경우 연동장치 외에 출입문 잠금장치가 있을 것
6) 6) 가동식 방책(출입문) 잠금장치는 가동식 방책(출입문)이 닫혀 있고, 잠겨 있는 상태에서만 로봇의 기동이 가능하게 할 것

(주) 레인보우로보틱스

본사 및 공장
대전광역시 유성구 엑스포로 339 번길 10-19

대표전화	042. 719. 8070
FAX	042. 719. 8071
대표메일	rainbow@rainbow-robotics.com
영업문의	sales@rainbow-robotics.com
홈페이지	www.rainbow-robotics.com