Start Guide

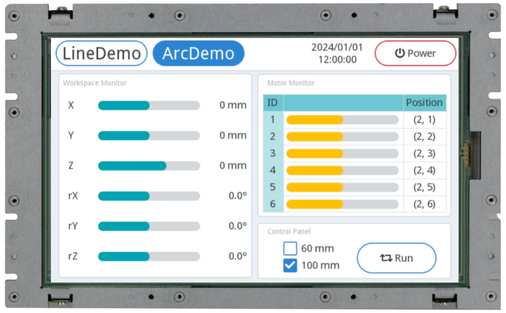
QEC-M-090T x AZD-KRED x OVR6048K1-V

EtherCAT Control + HMI Integration

ICOP × Orientalmotor

Date: 2025.11.17

(Version: v1r2)



QEC-M-090T
(HMI + EtherCAT MDevice)

EtherCAT

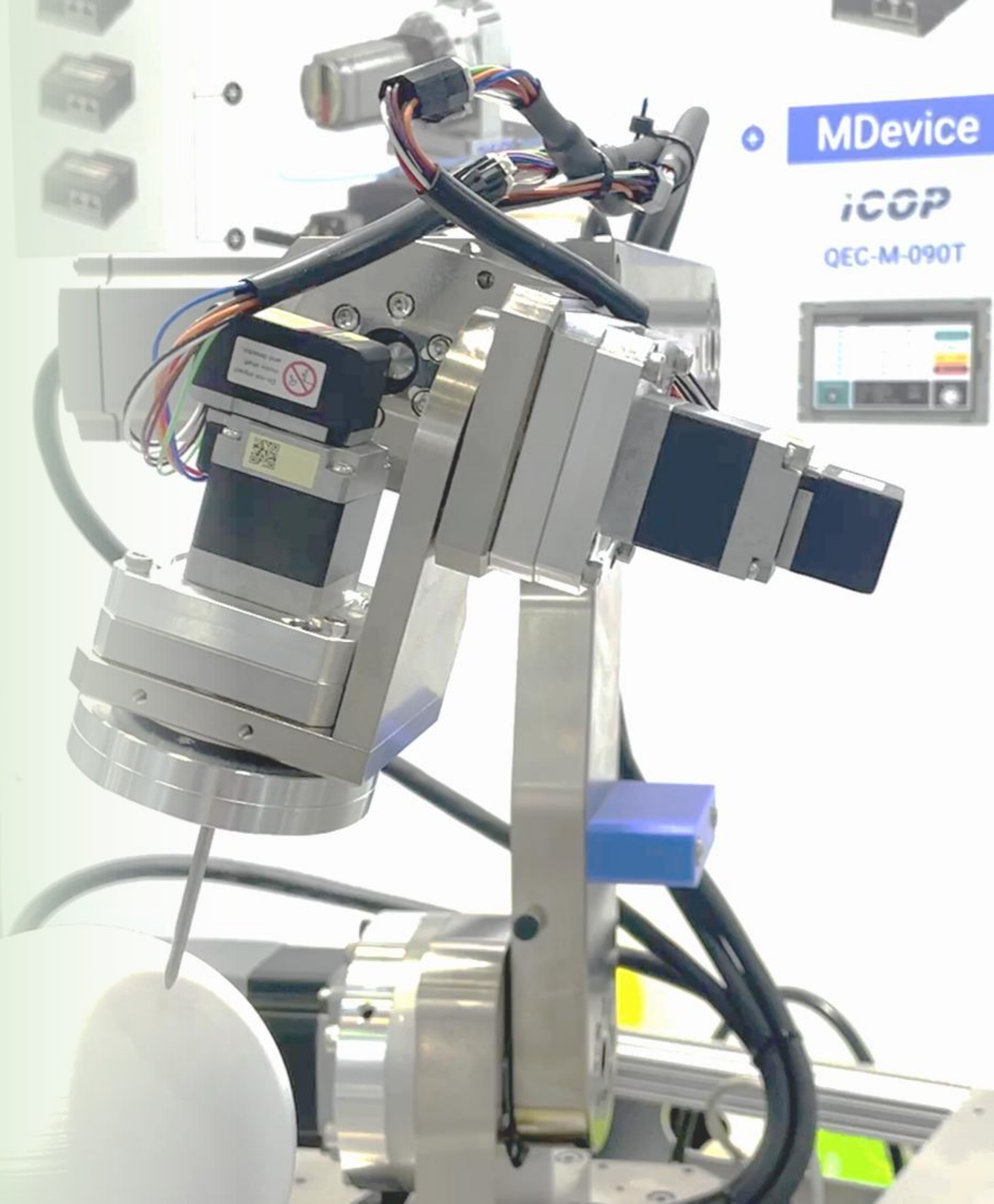
OVR6048K1-V
(6-Axis Robot, AZ Series)

Revision

Date	Version	Description
2025/10/10	Version1.0	New release.
2025/10/30	Version1.1	Add Kinematics Viewer parameters usage for the APIs.
2025/11/17	Version1.2	• Updated terminology: “Master/Slave” → “MDevice / SubDevice”. • Fixed typo on slide 59 (“down” → “done”). • Add 86Duino IDE 501+ introduction on slide 5.

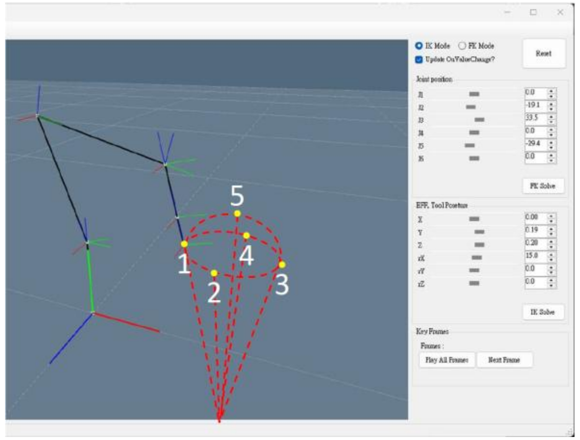
1. Project Introduction

Bill of Materials & Requirements



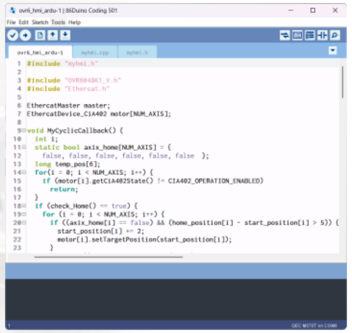
Project Introduction

We build a complete development flow for **OVR6048K1-V** using **QEC-M-090T** (EtherCAT MDevice + HMI) and **six AZD-KRED** drivers.
The system runs at **ECAT_SYNC** with **DC sync**, covering wiring and initial parameter templates, homing and safety, a beginner **Kinematics Test** in joint space, and a task-space **"Sphere Around"** demo.

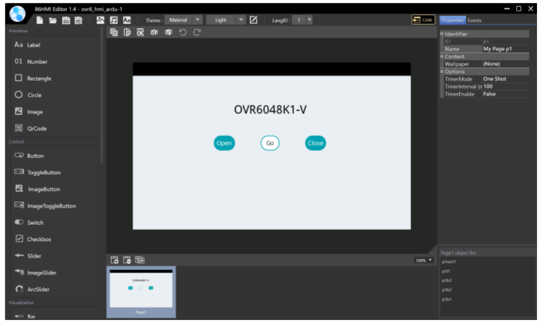


Kinematics Viewer

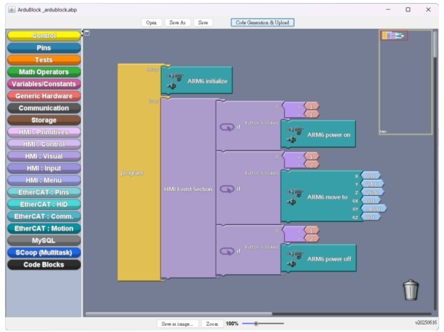
We also provide a low-code path—**ArduBlock** blocks tied to **86HMI**—so users can prototype motion, monitor PDOs, and validate results on **86Duino** quickly.



86Duino



86HMI



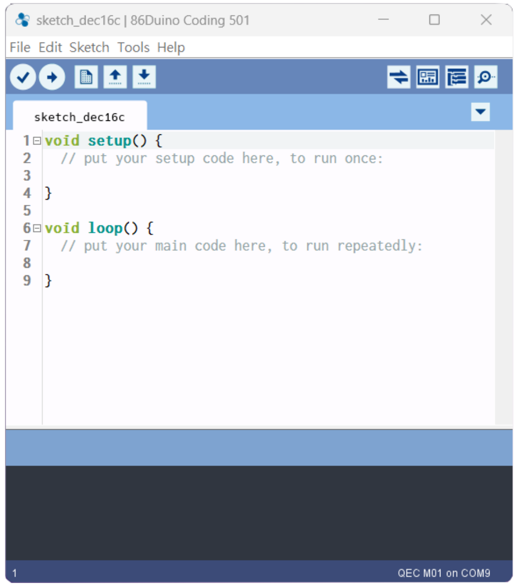
ArduBlock

About Software : 86Duino IDE 501+

The **86Duino integrated development environment (IDE)** makes it easy to write code and upload it to 86Duino boards. It runs on Windows, macOS, and Linux. The environment is written in Java and is based on Arduino IDE, Processing, DJGPP, and other open-source software.

86Duino IDE 501+ Logo

QEC.TW website



86Duino IDE 501+

What's new in 501+?

The new major release of 86Duino IDE 501+ is faster and more powerful.

It now supports a broader range of third-party EtherCAT SubDevices and introduces additional EthercatDevice classes, including a generic CiA-402 SubDevice class designed to control any EtherCAT servo drive compliant with the CiA-402 standard.

* Note: For the **Oriental Motor 6-axis OVR6048K1-V arm**, we provide a **special 86Duino IDE** build (includes model helpers and examples).

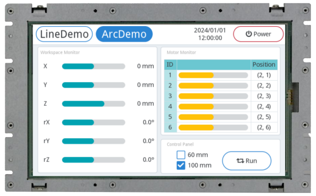
Bill of Materials & Requirements

Hardware

- **EtherCAT MDevice** : QEC-M-090T
- **EtherCAT Drivers** : AZD-KRED x6
- **Robot Arm & Mechanics** : OVR6048K1-V (Motors as below)
  - S: M1: DGB130R36-AZAKL
  - L: M2: AZM66MKH + CSG-20-100-2UH-LW-SP-B
  - U: M3: AZM66MKH + CSG-20-100-2UH-LW-SP-B
  - R: M4: AZM26AK + CSF-8-50-2UP-SP-A
  - Bp: M5: AZM24AK + CSF-8-50-2UP-SP-A
  - T: M6: AZM24AK + CSF-8-50-2UP-SP-A

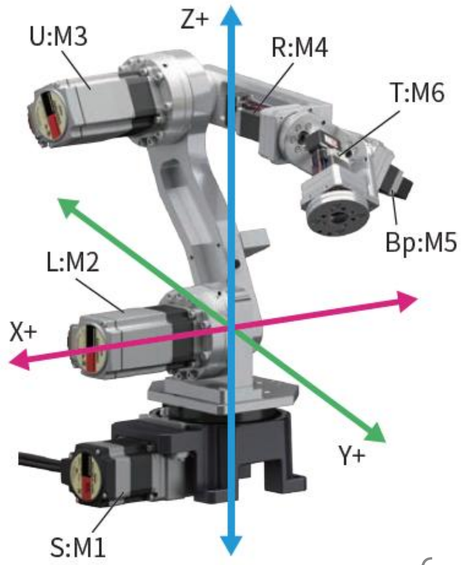
Software

- **86Duino IDE 501 Preview (Oriental Motor Robotics Arm Ver)**



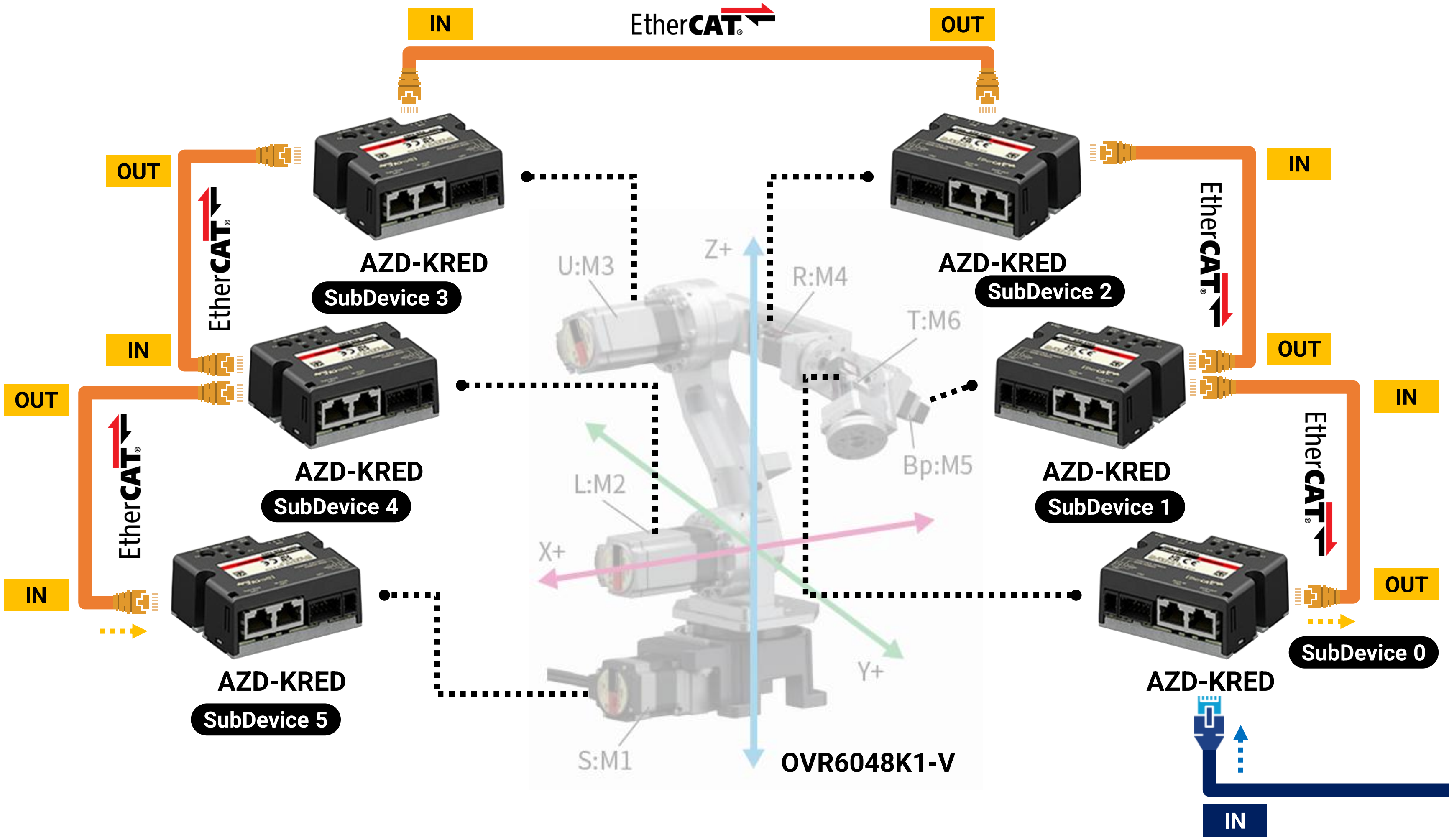
QEC-M-090T


AZD-KRED



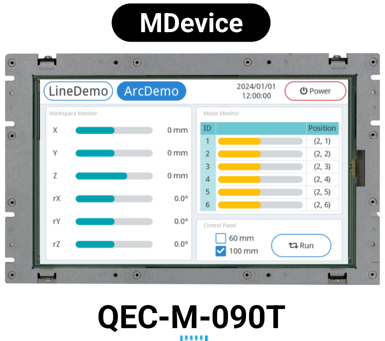
OVR6048K1-V

Wiring Diagram



Node Order

- S: **M1:** SubDevice 5
- L: **M2:** SubDevice 4
- U: **M3:** SubDevice 3
- R: **M4:** SubDevice 2
- Bp: **M5:** SubDevice 1
- T: **M6:** SubDevice 0



* Position-based addressing (fixed by cable order). Do not swap.

MDevice Reference | QEC-M-090T

- For **QEC-M-090T** specifications, wiring, features and safety, **always follow our official manuals:** https://www.qec.tw/product/qec-m-090t/
- This project uses **QEC-M-090T** (EtherCAT MDevice Open-frame 9” LCD).
- Product Advantages:
  - Integrated **EtherCAT MDevice on Real-time Dual-Core CPU**
  - **86Duino** Integrated Development Environment (IDE)
  - 9" TFT LCD with Resistive Touchscreen (800×480)
  - High Reliability SLC eMMC Storage with HDMI Output for HMI
  - Industrial Operating Temperature -20 to +70°C
  - **Full EtherCAT stack**, PDO/SDO, CiA-402 (PP/CSP/CSV), diagnostics & logging.
  - Built-in Voltage, Temperature, and System Status Monitoring

Robot Arm Reference | OVR6048K1-V

- For arm specifications (dimensions, payload, reach, limits), **always follow Oriental Motor official documentation**.
- See product page & manuals (CAD, mounting, safety notes):
  OVR 6-Axis Articulated Robot
  https://www.orientalmotor.com.tw/tw-zh/products/robots-controllers/ovr-6-axis-articulated-robot
- Please verify the following before integration:
  - **Payload / Reach**
  - **Joint Limits** (J1–J6, mechanical & soft)
  - **Home / Zero definition**
  - **Gear ratios & unit scaling** (deg ↔ pulse, vel/acc units)
  - **Tool-end interface** (flange, available I/O/wiring)
  - **Safety sections** (E-stop, keep-out zones, routing)

* Note: This deck shows a QEC integration workflow and **is not** an official spec. Always defer to the OEM manuals.

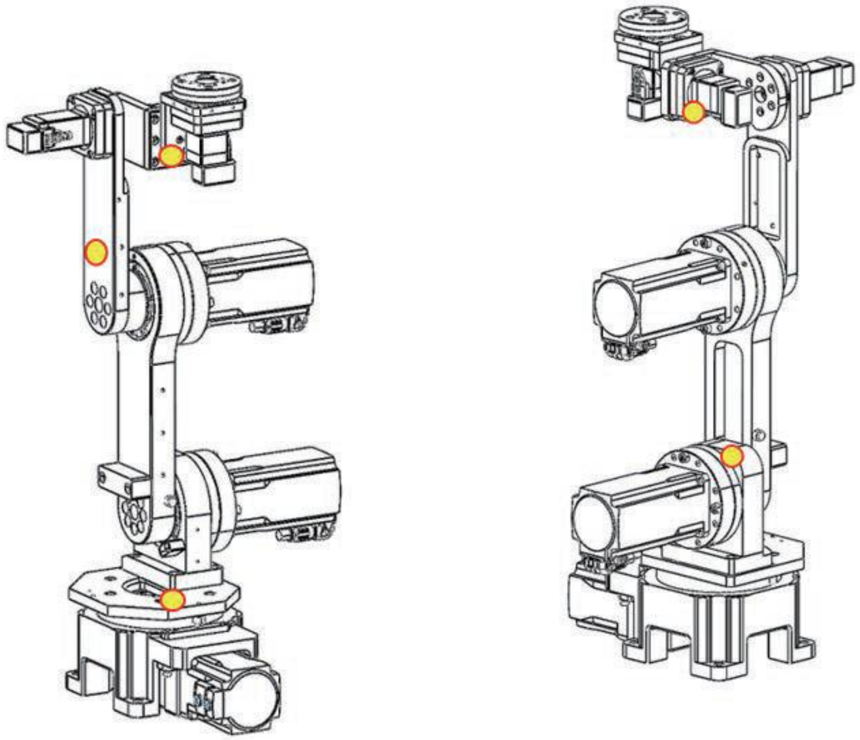
Driver Reference | AZD-KRED

- For AZ-mini driver specifications, wiring, features and safety, **always follow Oriental Motor's official manuals**: **AZ-mini Driver**
  https://www.orientalmotor.com.tw/tw-zh/products/alpha-step/az-mini-driver
- This project uses **AZD-KRED (EtherCAT)**.
- Please verify the following before integration:
  - **Model & Interface** (AZ-mini AZD-KRED EtherCAT)
  - **Power ratings** and grounding/shielding
  - Motor/brake compatibility and connectors
  - **ABZO absolute encoder** zeroing & backup method
  - **Units & Resolution** (electronic gearing, pulse/rev, vel/acc units & limits)
  - **Homing parameters** (method, speeds, offsets, in-position)
  - Environment & Mounting (temp/vibration, cable length, thermal)

* Note: Recommended driver for this deck: AZD-KRED (EtherCAT).

Zero Pose

- **Purpose**: Establish a canonical reference pose for all joints for coordinates/paths/API.
- **Reference**: Per OEM manual, this model provides five locating pin holes (φ4h7) to mechanically set the zero pose (see highlighted positions).
- **When**: Perform before first use, and after maintenance or driver replacement.
- **Procedure (recommended)**
  1. Switch to Manual/Teach at low speed and approach the pose.
  2. Insert locating pins at the five positions to confirm linkage alignment.
  3. Allow ±0.1° tolerance (configurable).
  4. Safety (Important): Do not input the driver's FREE signal while stopped or during motion. It cuts motor current and holding force; the robot may lose posture due to gravity and fall.



OVR6048K1 – Zero Pose

* Note: Follow the OEM manual for exact locations/limits.

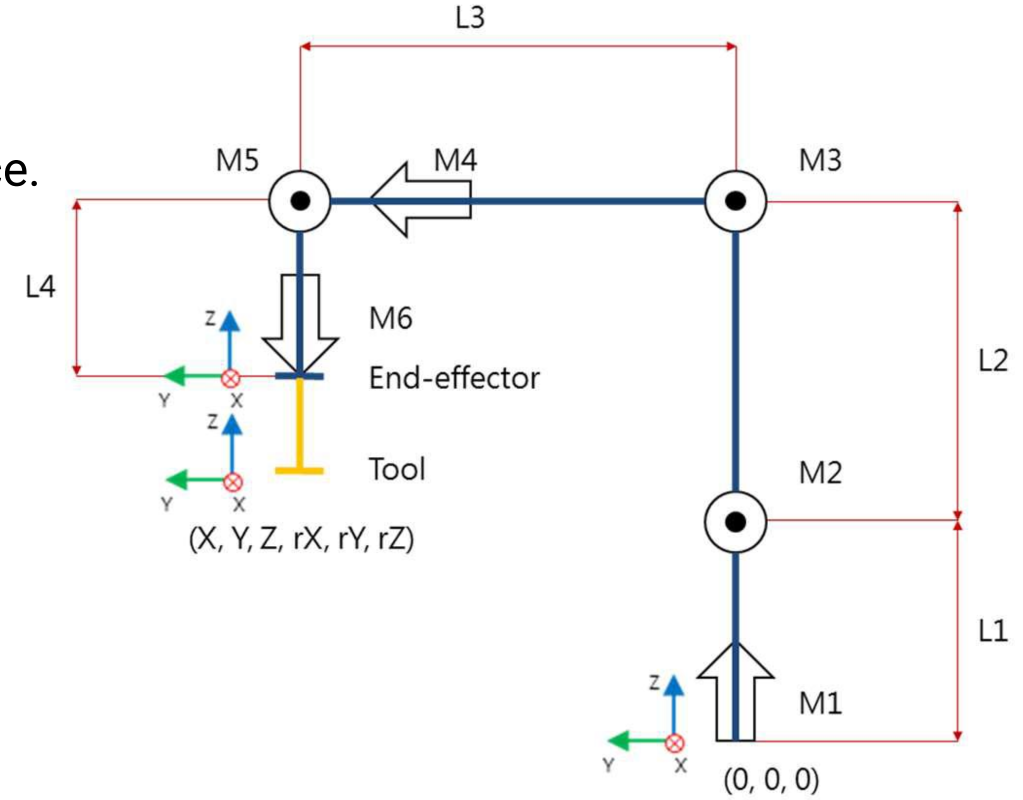
Zero Pose & Coordinate Conventions

Zero Pose

- **Base coordinate origin at (0,0,0)** on joint M1 center.
- **Standard pose** with J1–J6 = 0° is the global reference.
- **Orientation order (extrinsic):** rotate about world X → then world Y → then world Z
- **Composite matrix:**
  R_world = Rz(rZ) · Ry(rY) · Rx(rX)
  (applied in X→Y→Z order).

OVR6048K1 Zero-pose front view 1

OVR6048K1 Zero-pose front view 2



Zero Pose 2D view

* Note: Follow the OEM manual for exact locations/limits.

Resolution of Each Axis

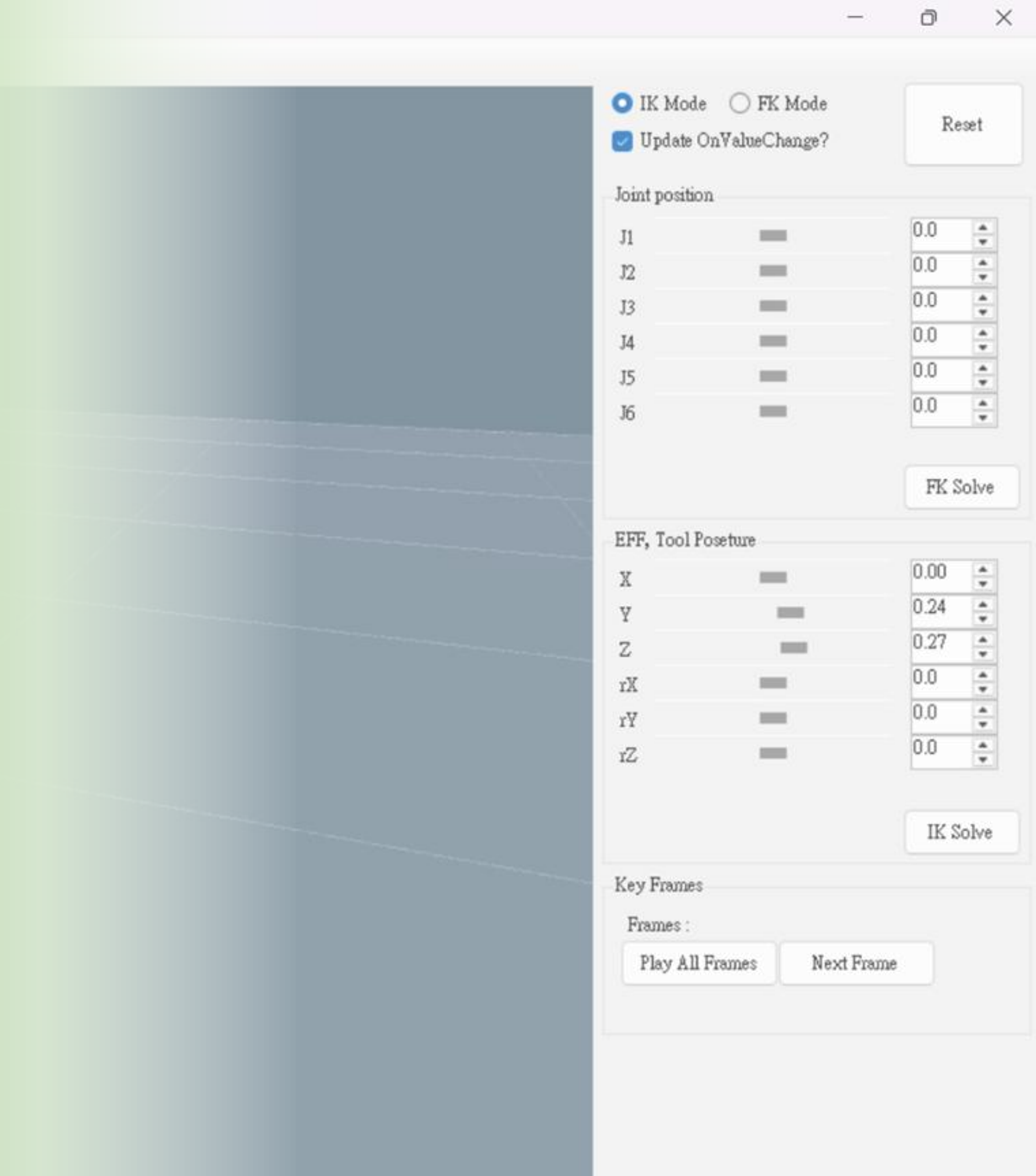
- The default resolution of the **AZD-KRED (driver) is 10,000**.

- The OVR6048K1-V axis models are as follows (see attached image)

Axis	Motor/Reducer	Total ratio	Steps / rev	Steps / deg	Deg / step
M1	DGB130R36-AZAKL	36	360,000	1,000.0	0.0010°
M2	AZM66MKH + CSG20-100	100[1]	1,000,000	2,777.78	0.00036°
M3	AZM66MKH + CSG20-100	100[1]	1,000,000	2,777.78	0.00036°
M4	AZM26AK + CSF8-50	50	500,000	1,388.89	0.00072°
M5	AZM24AK + CSF8-50	50	500,000	1,388.89	0.00072°
M6	AZM24AK + CSF8-50	50	500,000	1,388.89	0.00072°

2. Kinematics Viewer

OVR6048K1 Edition



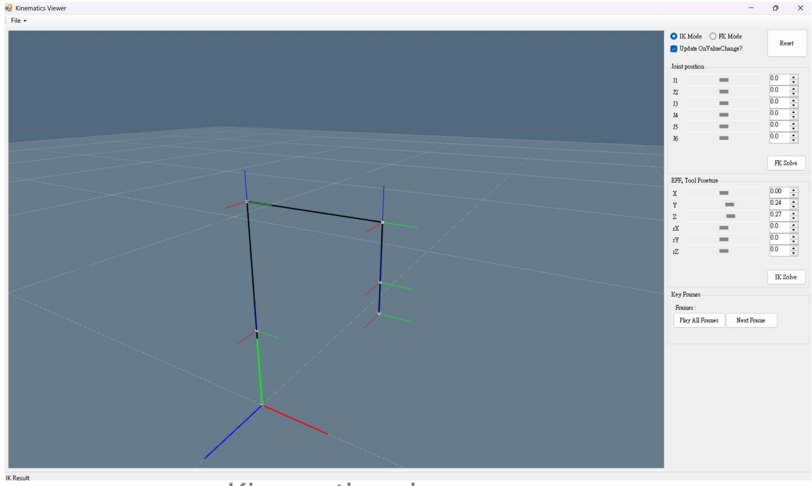
Kinematics Viewer Introduction - 1

What it is?

- A lightweight 3D simulator pre-configured with **OVR6048K1** link lengths, zero pose, and axes.
- Supports **FK** (forward) and **IK** (inverse) kinematics with live visualization.

Why use it

- Validate poses, paths, and singularities **before** running the real robot.
- Generate waypoints/timings and hand them off to the QEC program/HMI.



Kinematics viewer

Kinematics Viewer Introduction - 2

Key features

- Modes: IK / FK toggle, optional Update **OnValueChange** for instant recompute.
- Pose I/O: edit **J1–J6 or TCP (X, Y, Z, rX, rY, rZ)** and solve either way.
- Keyframes: step with **Next Frame** or **Play All Frames**.
- Import path: load testPath.txt to simulate a full motion timeline.

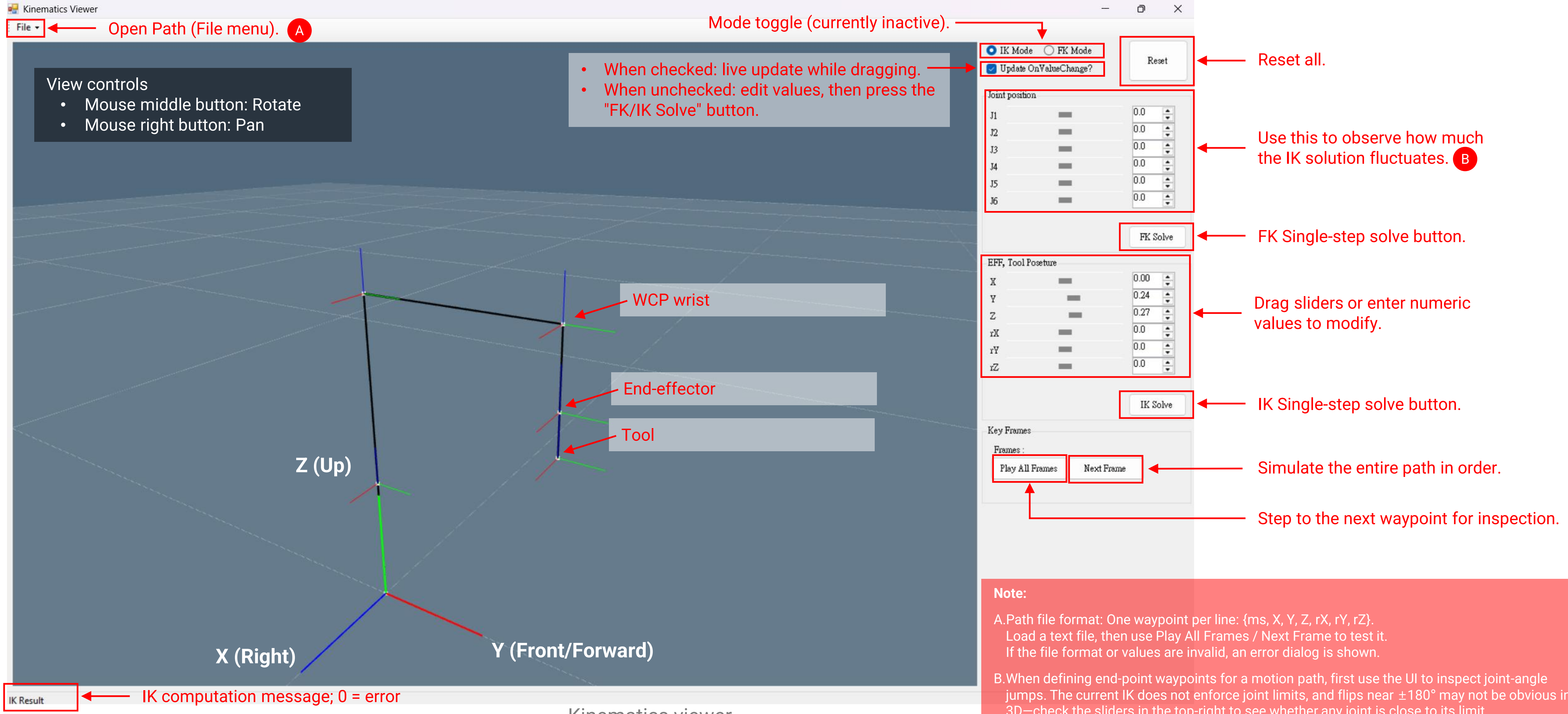
Conventions

- Extrinsic rotation order **X → Y → Z (world axes).**
- Zero pose aligned to the OVR6048K1 reference; units: **position m/mm**, angles **deg**.

Quick controls

- Mouse: left-drag rotate, middle-drag pan, wheel zoom.
- Actions: **FK Solve / IK Solve, Reset**.

Kinematics Viewer UI Functions



Note:

A. Path file format: One waypoint per line: {ms, X, Y, Z, rX, rY, rZ}.
Load a text file, then use Play All Frames / Next Frame to test it.
If the file format or values are invalid, an error dialog is shown.

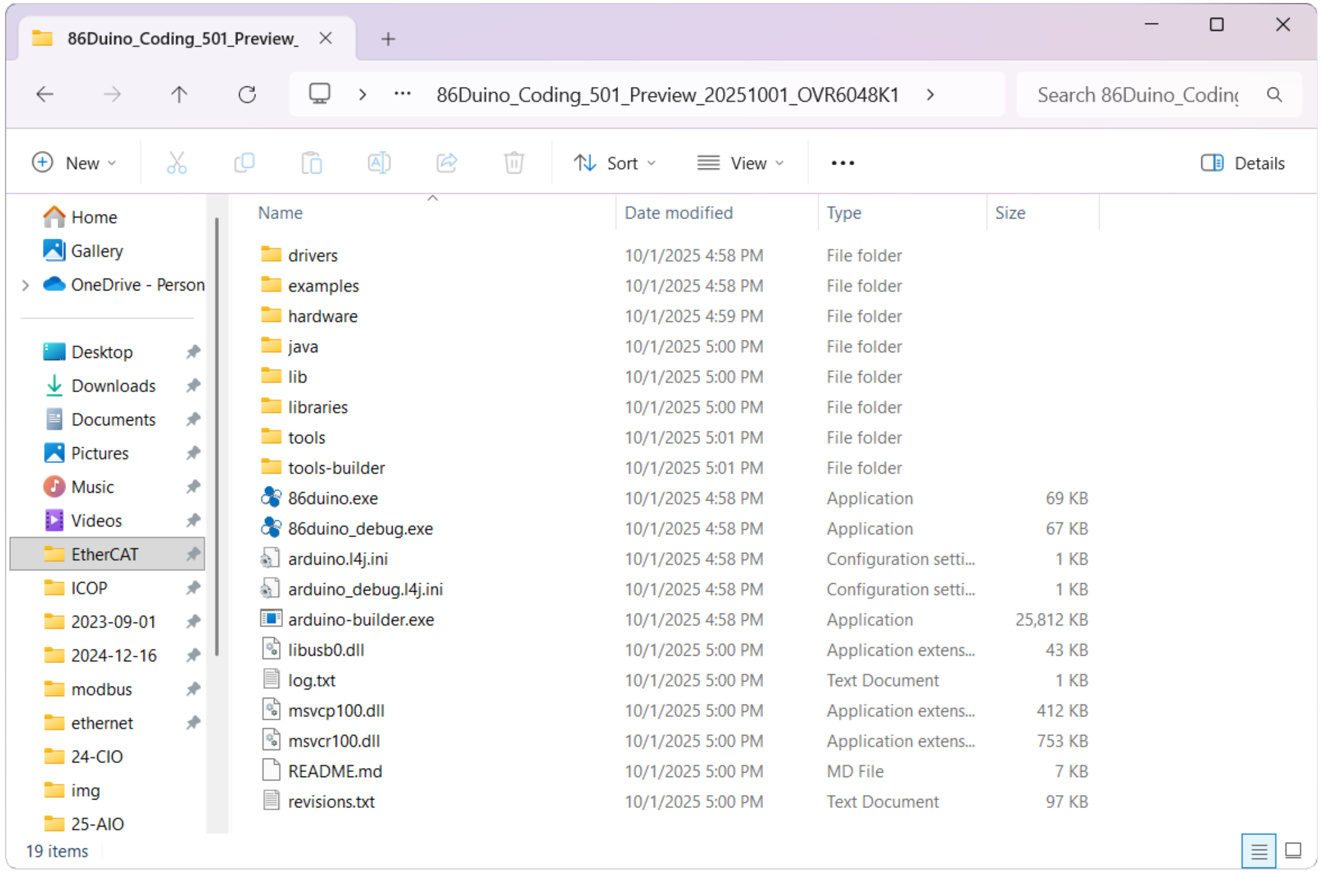
B. When defining end-point waypoints for a motion path, first use the UI to inspect joint-angle jumps. The current IK does not enforce joint limits, and flips near ±180° may not be obvious in 3D—check the sliders in the top-right to see whether any joint is close to its limit.

Kinematics viewer

Launching the Kinematics Viewer - 1

Step 1 — Open the IDE folder

- Go to your 86Duino IDE directory (e.g., …\86Duino_Coding_501_Preview_20251001_OVR6048K1).

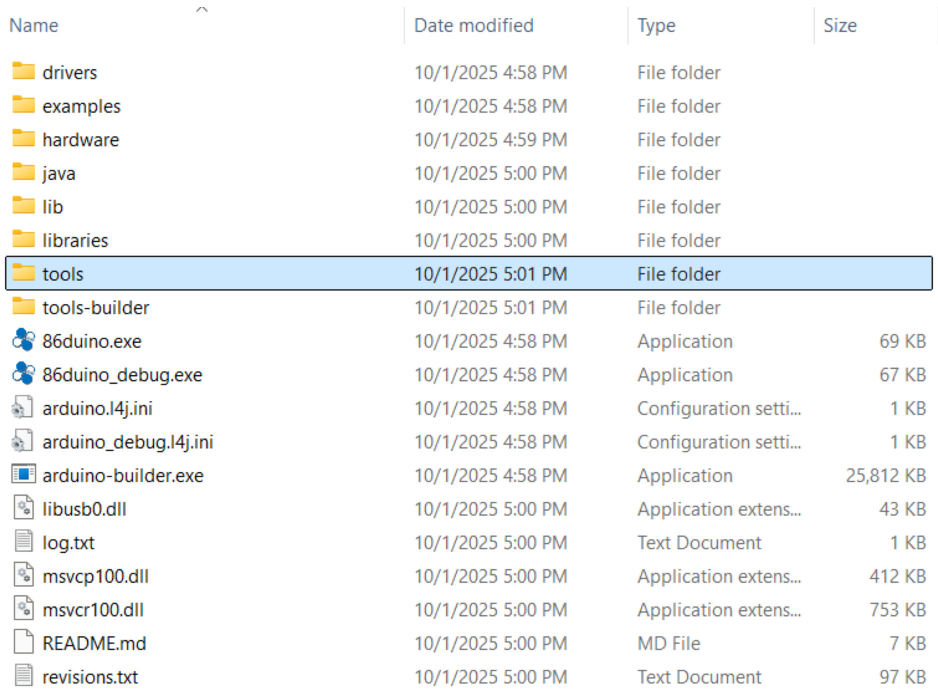


Kinematics viewer file path

Launching the Kinematics Viewer - 2

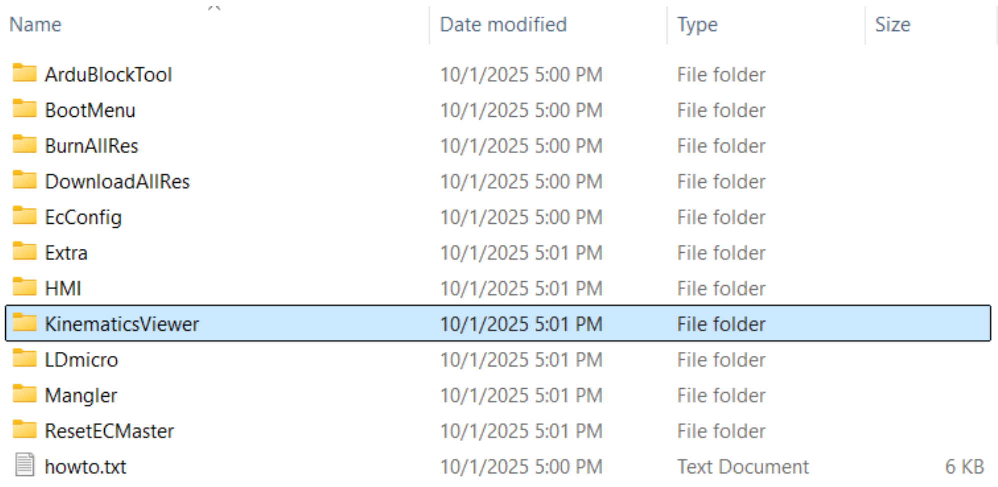
Step 2 — Navigate to the tool

- Open: tools → KinematicsViewer.



Name	Date modified	Type	Size
drivers	10/1/2025 4:58 PM	File folder	
examples	10/1/2025 4:58 PM	File folder	
hardware	10/1/2025 4:59 PM	File folder	
java	10/1/2025 5:00 PM	File folder	
lib	10/1/2025 5:00 PM	File folder	
libraries	10/1/2025 5:00 PM	File folder	
tools	10/1/2025 5:01 PM	File folder	
tools-builder	10/1/2025 5:01 PM	File folder	
86duino.exe	10/1/2025 4:58 PM	Application	69 KB
86duino_debug.exe	10/1/2025 4:58 PM	Application	67 KB
arduino.l4j.ini	10/1/2025 4:58 PM	Configuration setti...	1 KB
arduino_debug.l4j.ini	10/1/2025 4:58 PM	Configuration setti...	1 KB
arduino-builder.exe	10/1/2025 4:58 PM	Application	25,812 KB
libusb0.dll	10/1/2025 5:00 PM	Application extens...	43 KB
log.txt	10/1/2025 5:00 PM	Text Document	1 KB
msvcp100.dll	10/1/2025 5:00 PM	Application extens...	412 KB
msvcr100.dll	10/1/2025 5:00 PM	Application extens...	753 KB
README.md	10/1/2025 5:00 PM	MD File	7 KB
revisions.txt	10/1/2025 5:00 PM	Text Document	97 KB

Kinematics viewer file path 2



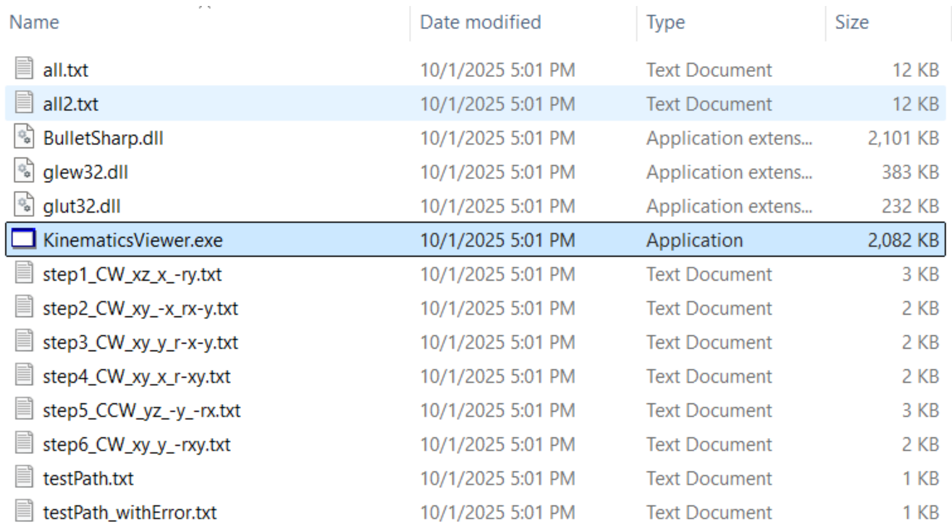
Name	Date modified	Type	Size
ArduBlockTool	10/1/2025 5:00 PM	File folder	
BootMenu	10/1/2025 5:00 PM	File folder	
BurnAllRes	10/1/2025 5:00 PM	File folder	
DownloadAllRes	10/1/2025 5:00 PM	File folder	
EcConfig	10/1/2025 5:00 PM	File folder	
Extra	10/1/2025 5:01 PM	File folder	
HMI	10/1/2025 5:01 PM	File folder	
KinematicsViewer	10/1/2025 5:01 PM	File folder	
LDmicro	10/1/2025 5:01 PM	File folder	
Mangler	10/1/2025 5:01 PM	File folder	
ResetECMaster	10/1/2025 5:01 PM	File folder	
howto.txt	10/1/2025 5:00 PM	Text Document	6 KB

Kinematics viewer file path 3

Launching the Kinematics Viewer - 3

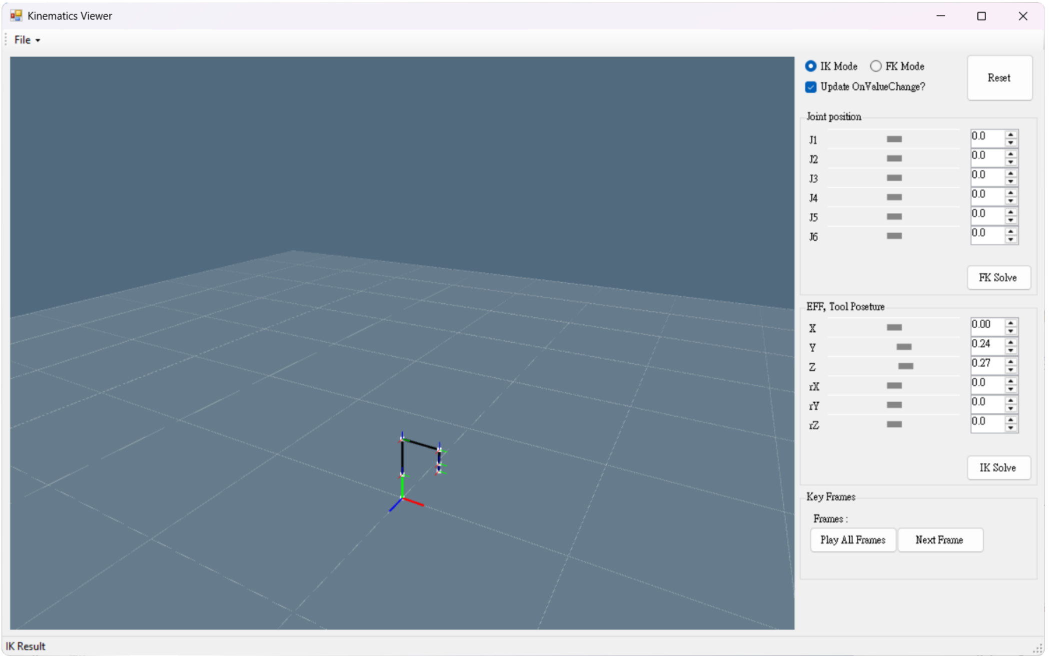
Step 3 — Run the viewer

- Double-click **KinematicsViewer.exe**.



Kinematics viewer file path 4

Kinematics viewer 2

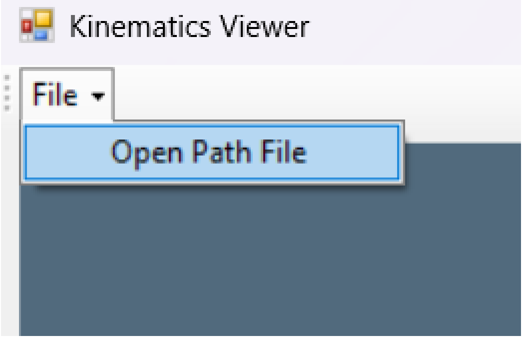


You'll see the executable alongside several sample .txt files.

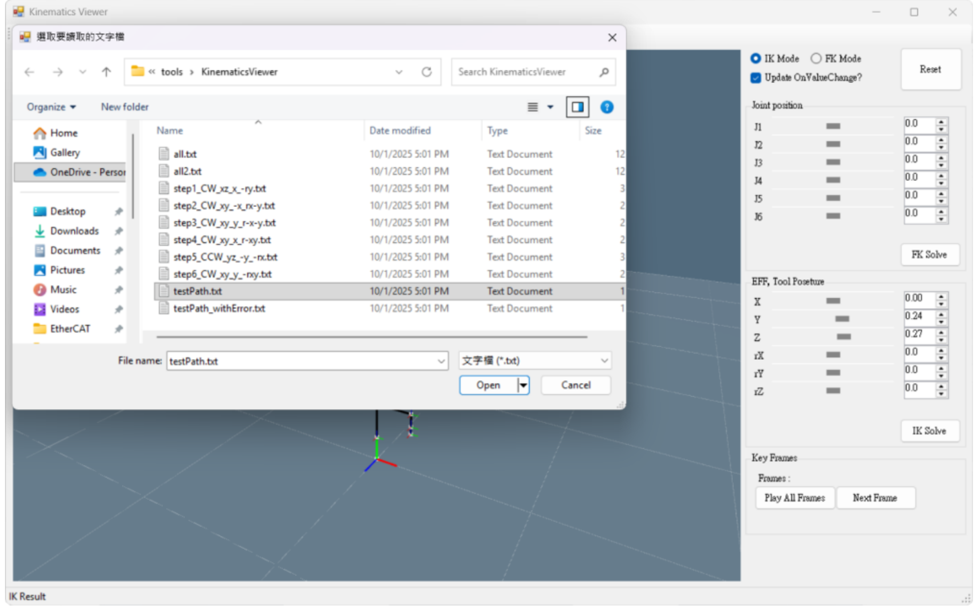
Launching the Kinematics Viewer - 4

Step 4 — Load a motion path

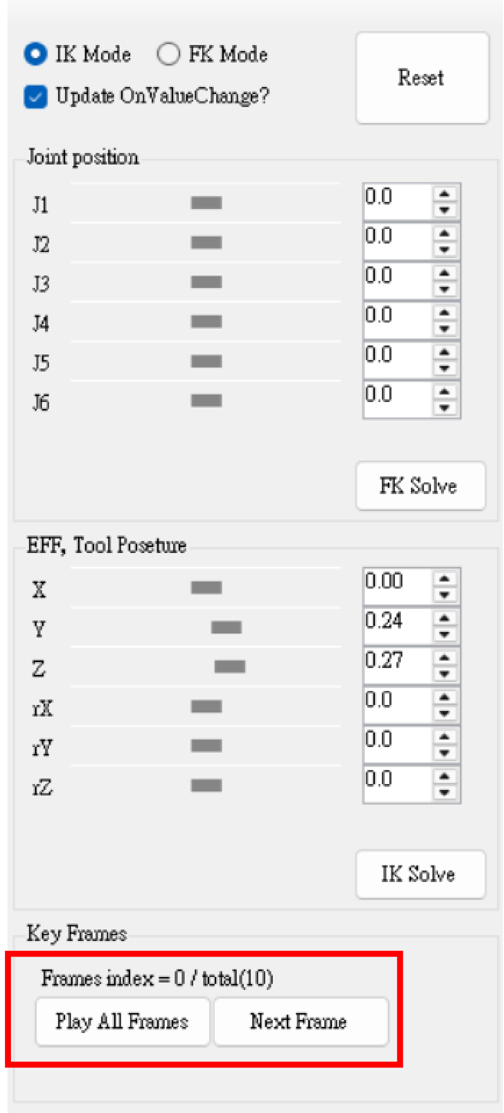
- In the viewer: **File** → **Import/Open Path…** → choose **testPath.txt** *(or any of the other sample files such as all.txt, all2.txt, step1_…txt …).*
- Use **Play All Frames** / **Next Frame** to simulate the trajectory.



Kinematics viewer – file open 1


Kinematics viewer – file open 2

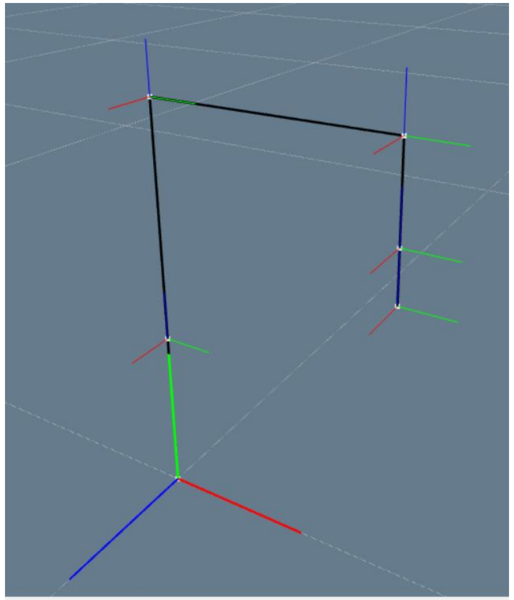
Kinematics viewer function



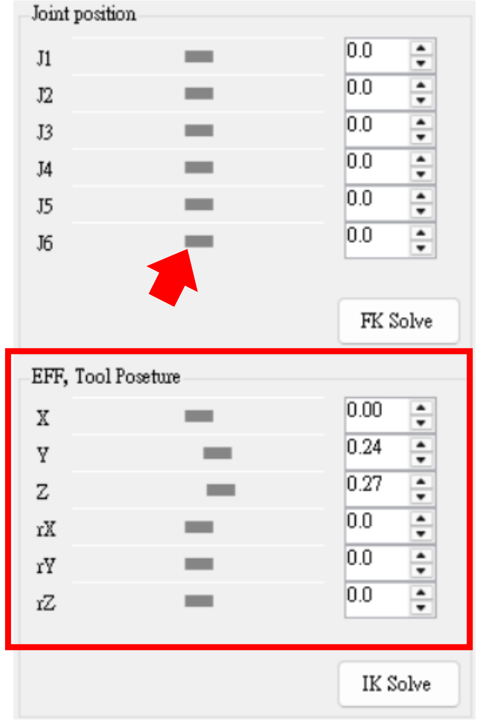
Launching the Kinematics Viewer - 5

Step 5 — Change the value

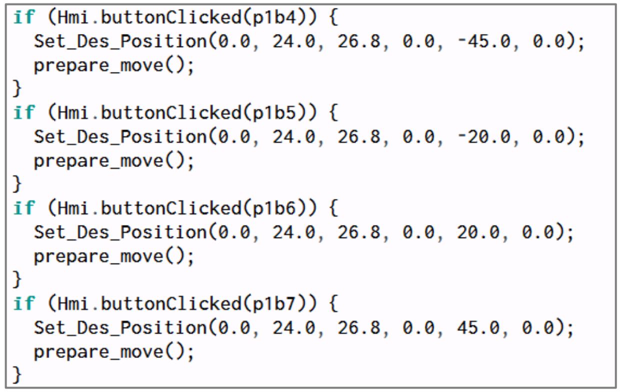
In the viewer, the user can drag the bar or the input value in the "**Joint position**" or "**EFF, Tool Poseture**" areas to simulate robotics in the views window. After finishing, users can use the "**EFF, Tool Poseture**" area value and input to the corresponding API functions to control the arm to the same position as in the Kinematics viewer.



Viewer Windows

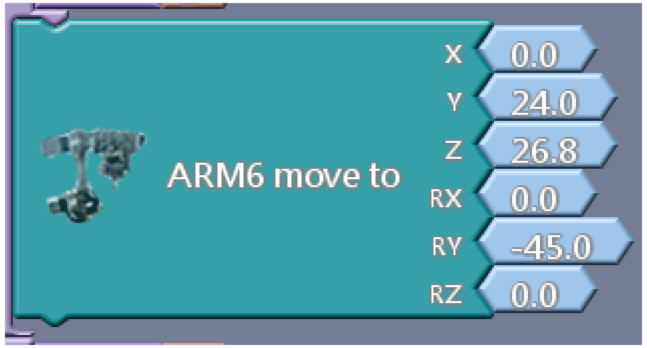


Kinematics viewer function



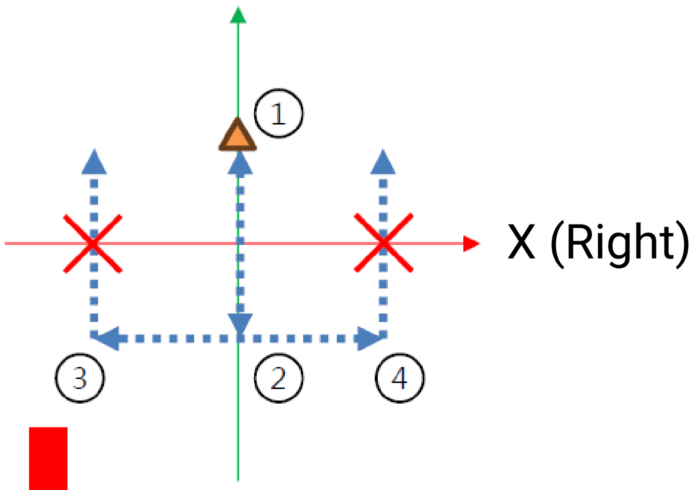
```
if (Hmi.buttonClicked(p1b4)) {
  Set_Des_Position(0.0, 24.0, 26.8, 0.0, -45.0, 0.0);
  prepare_move();
}
if (Hmi.buttonClicked(p1b5)) {
  Set_Des_Position(0.0, 24.0, 26.8, 0.0, -20.0, 0.0);
  prepare_move();
}
if (Hmi.buttonClicked(p1b6)) {
  Set_Des_Position(0.0, 24.0, 26.8, 0.0, 20.0, 0.0);
  prepare_move();
}
if (Hmi.buttonClicked(p1b7)) {
  Set_Des_Position(0.0, 24.0, 26.8, 0.0, 45.0, 0.0);
  prepare_move();
}
```

Kinematics Example Code

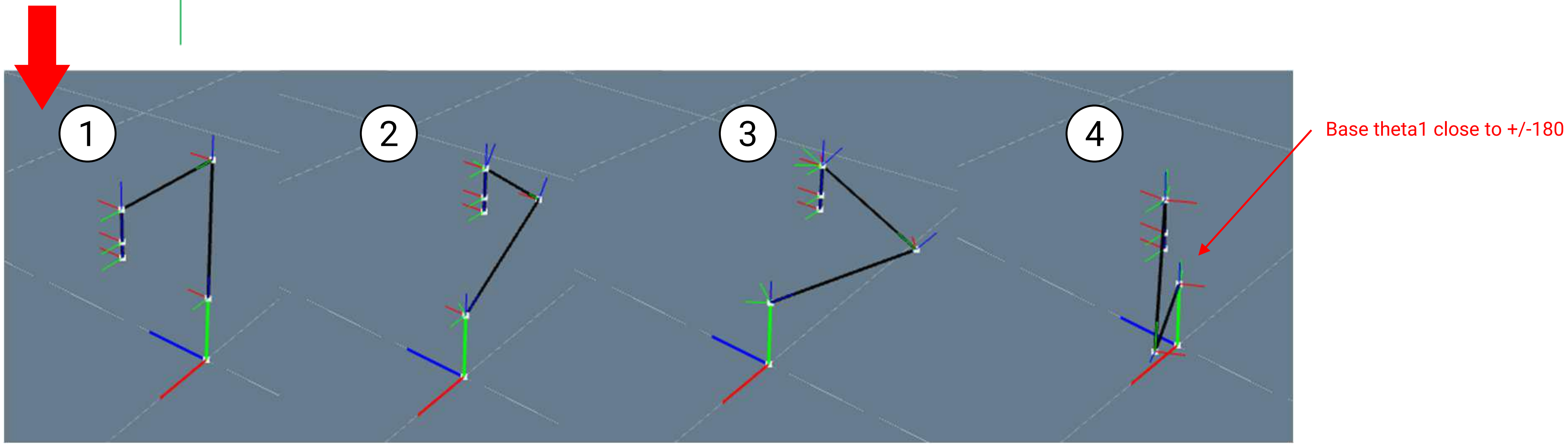


ArduBlock Moving the Robotics Arm Block

Path planning needs to avoid - 1

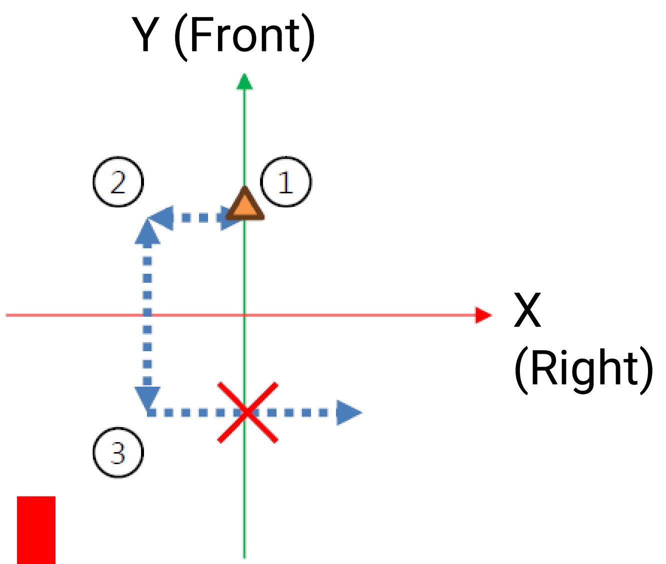


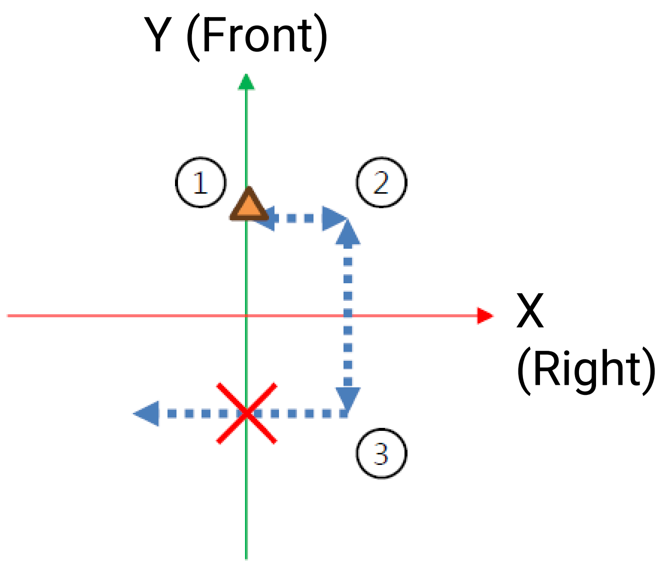
*Note: Some movement paths will cause the base to rotate close to its limits. Cross-quadrant paths marked with an X must be avoided and returned to the quadrant of the original path.



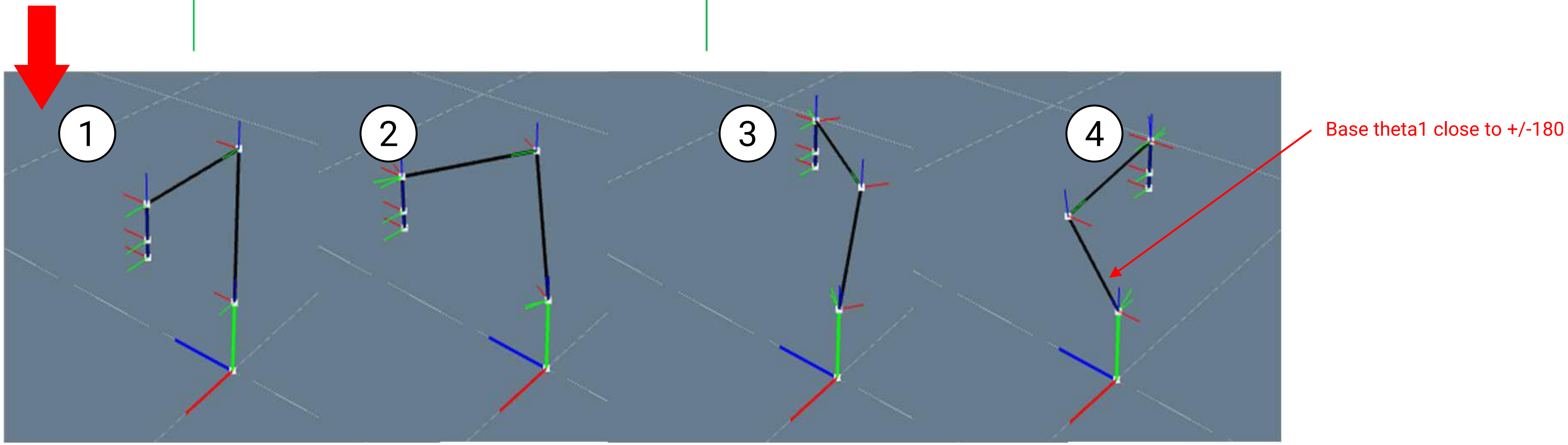
Kinematics avoid path -1

Path planning needs to avoid - 2



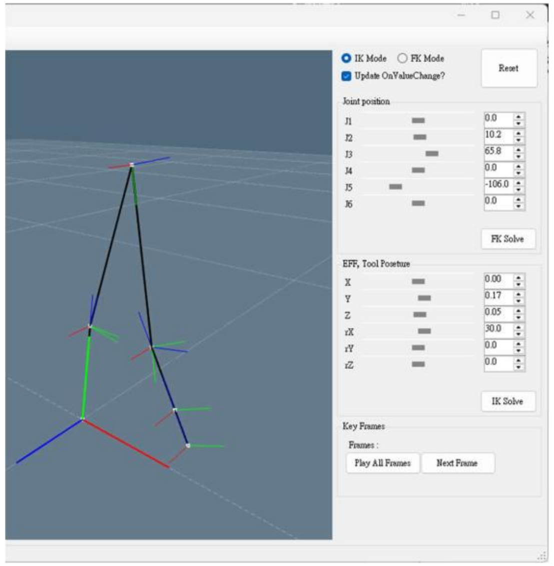


*Note: Some movement paths will cause the base to rotate close to its limits. Cross-quadrant paths marked with an X must be avoided and returned to the quadrant of the original path.

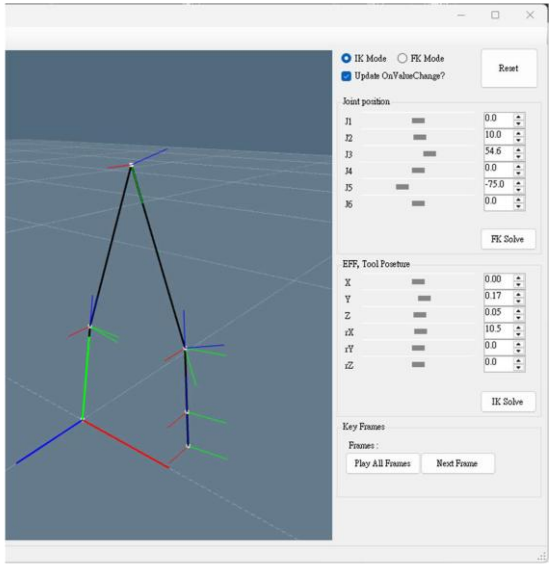


Kinematics avoid path - 2

Path planning needs to avoid - 3



Kinematics avoid path - 3



Kinematics avoid path - 4

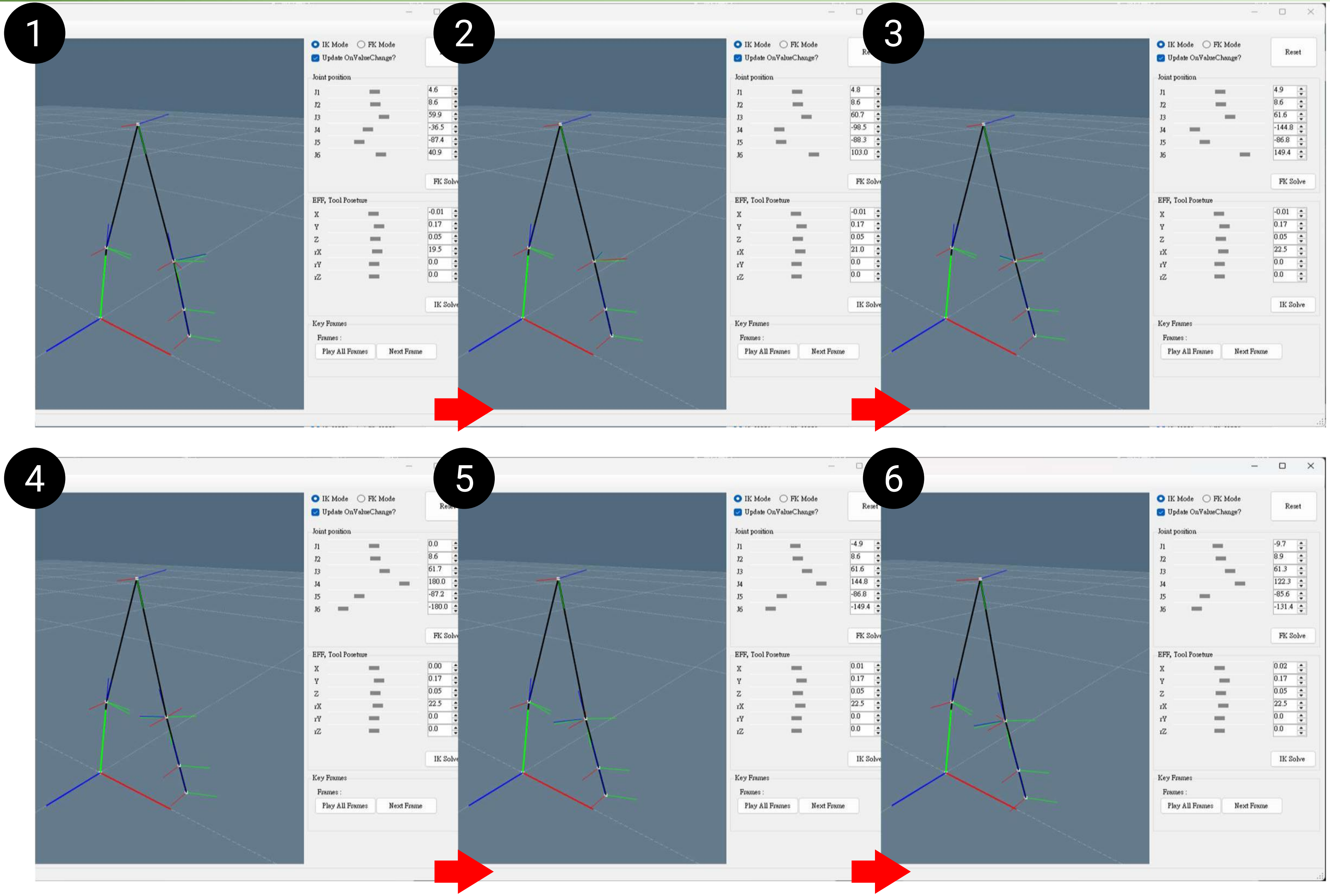
Check the wrist rotation. On the X=0 plane, **Joint 5** is fine when close to -90 degrees.

However, if X deviates slightly from the X=0 plane (for example, X=-0.01), moving J5 closer to +/-90 degrees will cause Theta 4/6 to move toward the edge because Theta 4 minimizes the variation to reduce jitter.

At this point, if the end position X is shifted across quadrants (for example, X=-0.01   0.01), Theta 4 and Theta 6 will move across +/-180 degrees. Therefore, avoid performing similar movements when J5 is close to +/-90 degrees.

(Continued on the next page.)

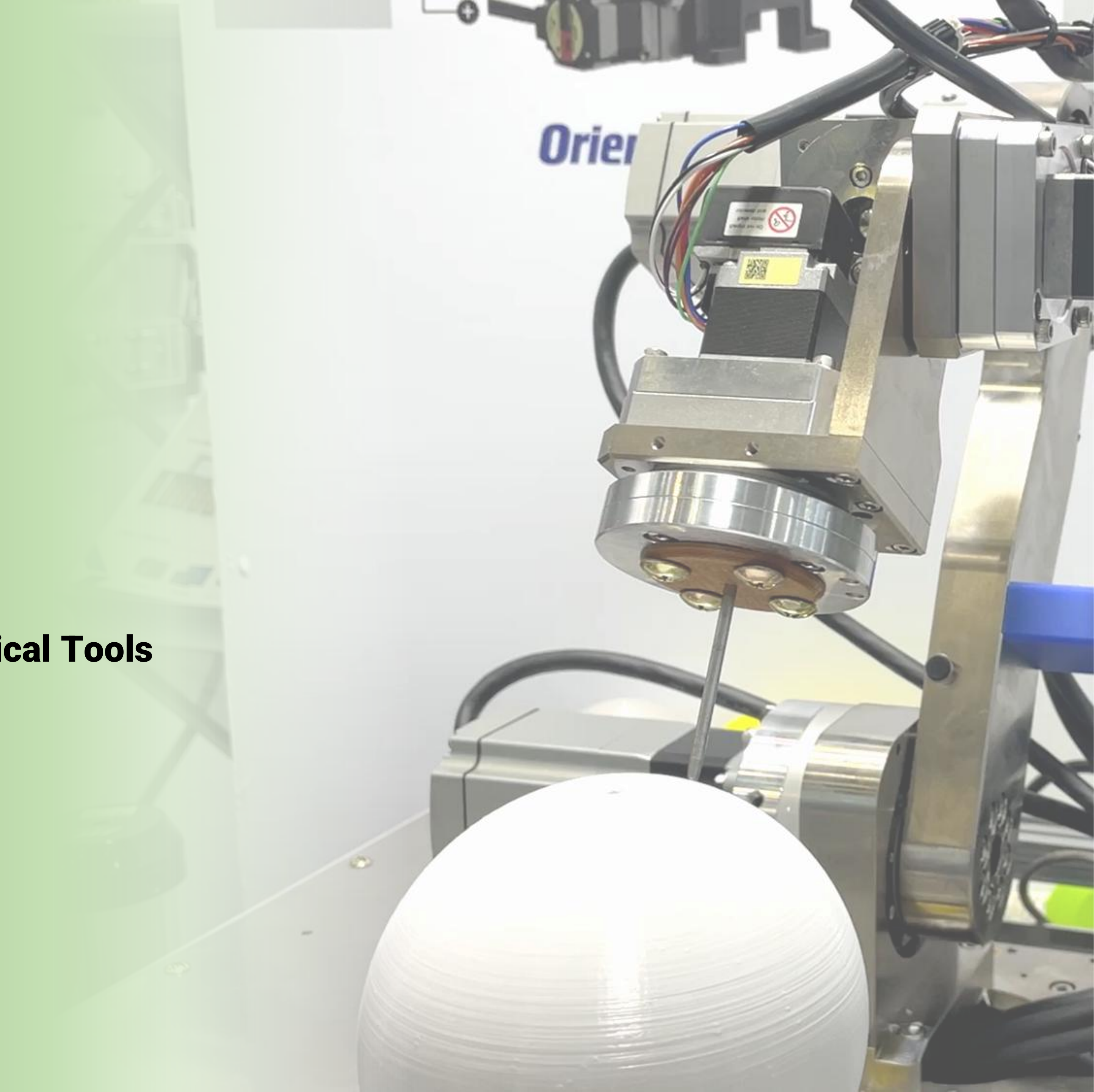
Path planning needs to avoid - 4



Kinematics avoid path - 5

3. How to use?

API Introduction, Coding Method, and Graphical Tools

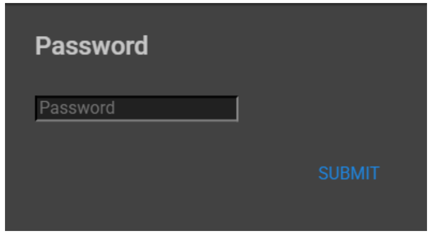


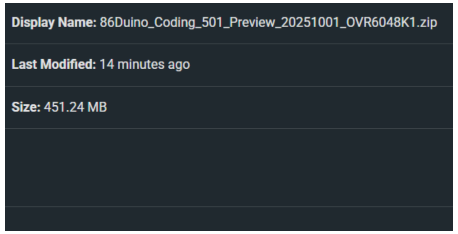
Software Overview & Download

For the **Oriental Motor's Robotics Arm (6-axis),** we release a **special 86Duino IDE version**. It supports **motion**, **kinematics**, and the **viewer**, which are specific to the OVR6048K1-V.

Download 86duino IDE from https://ftp.dmp.com.tw/share/Ch_rBIbF

- Password: icop
- Filename: 86Duino_Coding_501_Preview_20251001_OVR6048K1.zip





The new major release of the **86Duino IDE 501** is faster and more powerful than ever! It now supports a broader range of third-party EtherCAT SubDevices and introduces additional EthercatDevice classes, including a generic CiA 402 EtherCAT SubDevice class designed to control any EtherCAT servo drive compliant with the CiA 402 standard.

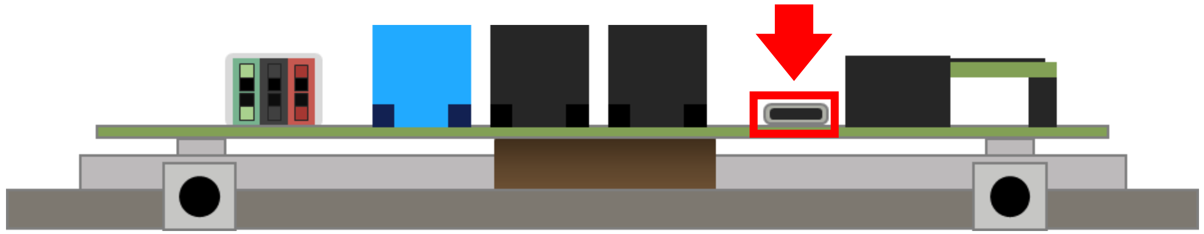
- **EtherCAT Library API User Manual:** https://www.qec.tw/ethercat/api/ethercat-library-api-user-manual/

86Duino IDE 501

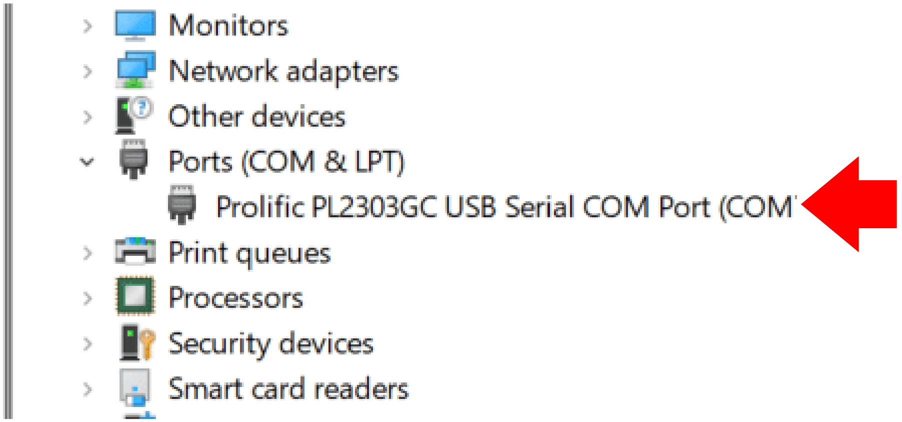
Development Environment Setup - 1

Follow the steps below to set up the environment:

1. Connect the QEC-M-090T to your PC via a USB Type-C to USB cable (86Duino IDE installed).
2. Turn on the QEC power.
3. Open “**Device Manager**” (select in the menu after pressing Win+X) ->” **Ports (COM & LPT)**” in your PC and expand the ports; you should see that the “**Prolific PL2303GC USB Serial COM Port (COMx)**” is detected; if not, you will need to install the required drivers.
(For Windows PL2303 driver, you can download here)

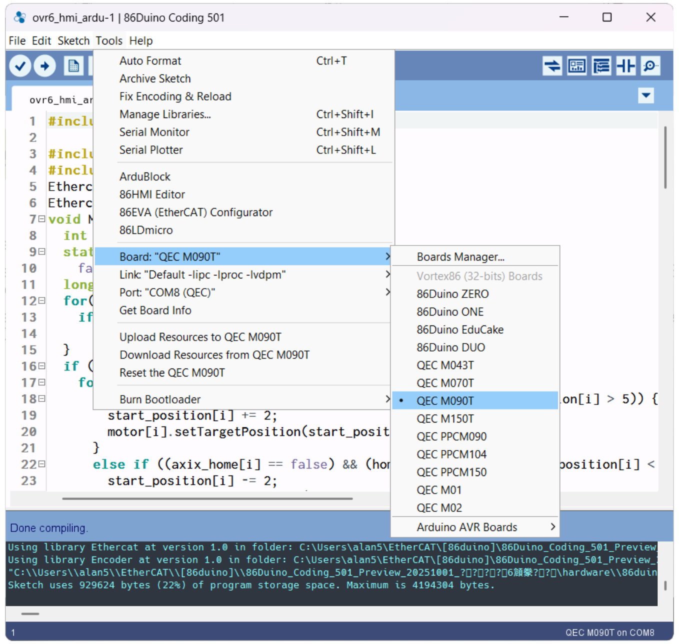


QEC-M Open frame board side view

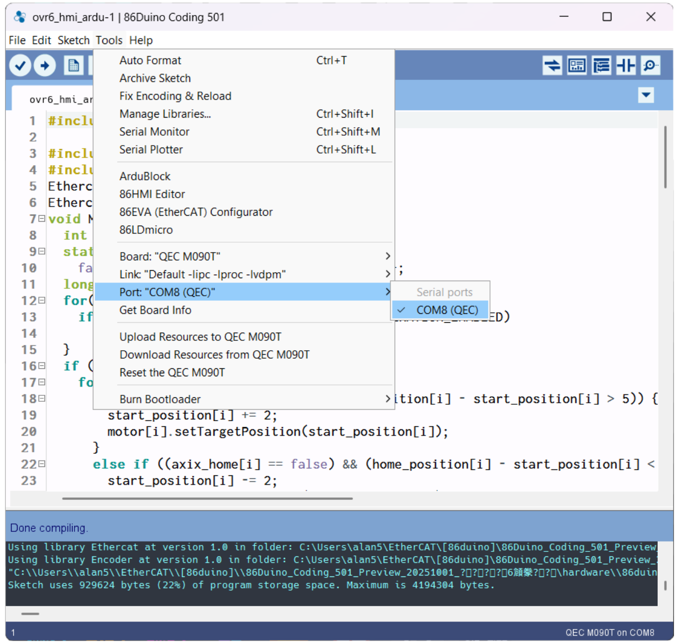


Development Environment Setup - 2

4. Open the 86Duino IDE.
5. Select the correct board: In the IDE's menu, select “**Tools**” > “**Board**” > “**QEC-M090T**” (or the QEC MDevice model you use).
6. Select Port: In the IDE's menu, select “**Tools**” > “**Port**” and select the USB port to connect to the QEC MDevice (in this case, COM8 (QEC)).



Tool - Board



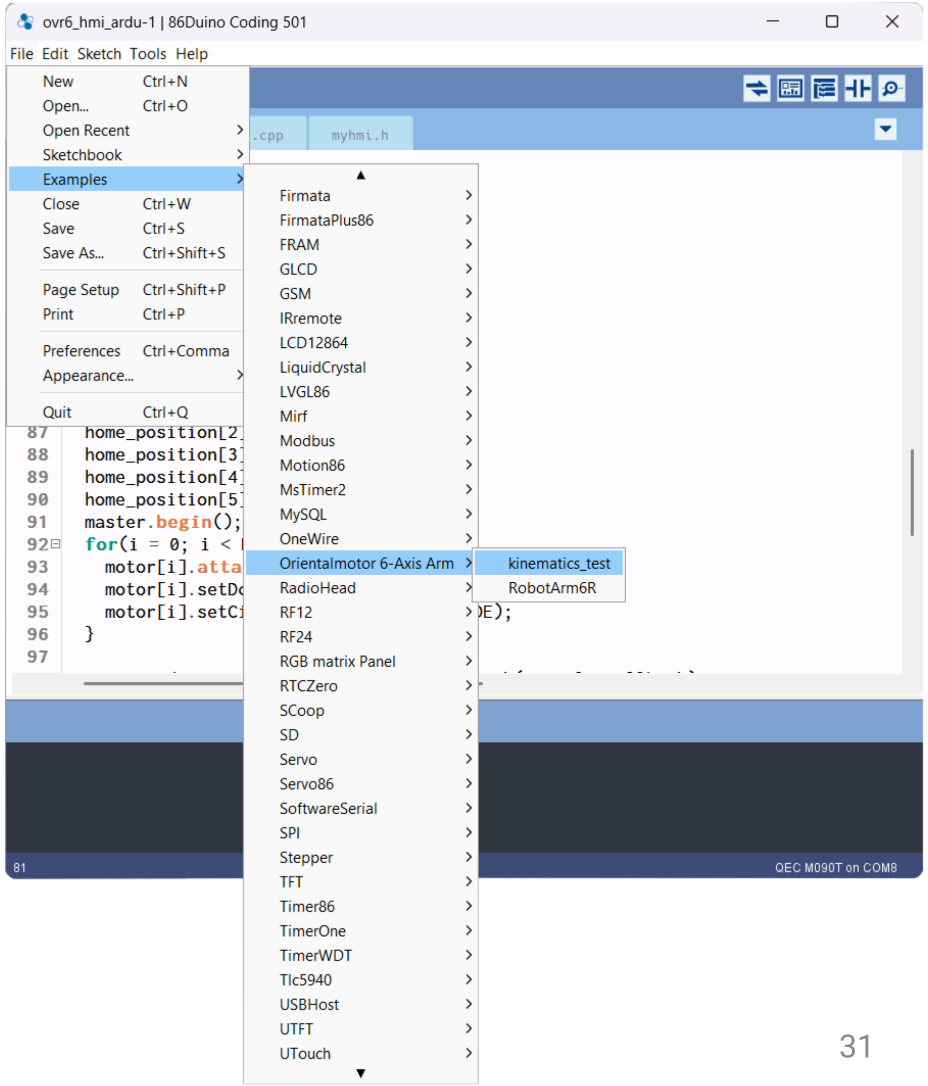
Tool - Port

Example: Kinematics Test - 1

Step 1 — Open the Example

- After selecting “**QEC-M090T**” under "Tools ▸ Board” and the correct Port, you can find the example set
- Open “File ▸ Examples ▸ Orientalmotor 6-Axis Arm ▸ kinematics_test”.

86Duino Examples



Example: Kinematics Test - 2

Step 2 — View Code & Open HMI

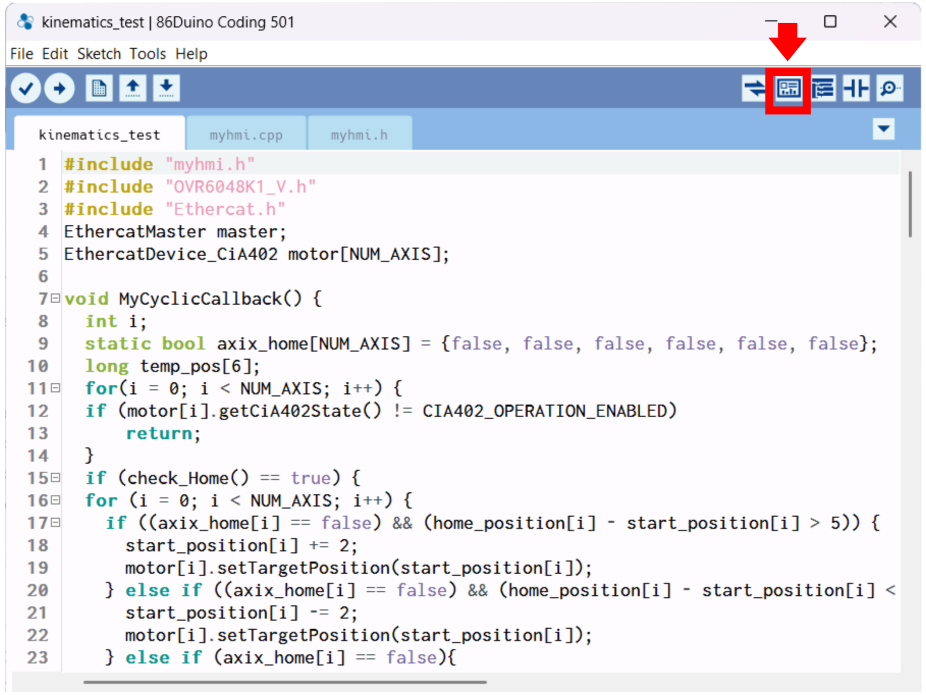
- Opening kinematics_test loads the sketch (right). It uses three libraries:
    - **myhmi.h** — HMI helpers
    - **OVR6048K1_V.h** — robot motion helpers
    - **Ethercat.h** — EtherCAT protocol
- This example uses the following settings:
    - EtherCAT Cycle Time: **250 microseconds (μs)**.
    - EtherCAT Mode: **ECAT_SYNC**.
    - Distributed Clock (DC): **Enable (250 μs).**
    - CiA-402 mode: **CSP (Cyclic Synchronous Position).**

```
master.begin();
for(i = 0; i < NUM_AXIS; i++) {
  motor[i].attach(i, master);
  motor[i].setDc(cycle_time*10);
  motor[i].setCiA402Mode(CIA402_CSP_MODE);
}

error_code = master.attachRTCyclicCallback(MyCyclicCallback);
if (error_code != 0) {
 printf("error : %d\n", error_code);
}
master.start(cycle_time*10, ECAT_SYNC); // cycle_time = 25000; // nanosecond
```

Kinematics_test.ino Code about EtherCAT

- Click the 86HMI icon (upper-right) to launch the HMI Editor.



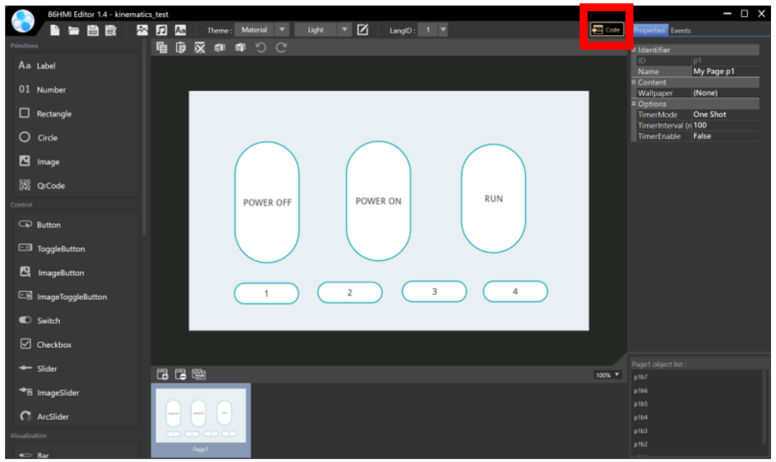
Kinematics_test.ino

Example: Kinematics Test - 3

Step 3 — 86HMI Layout & Code Generation

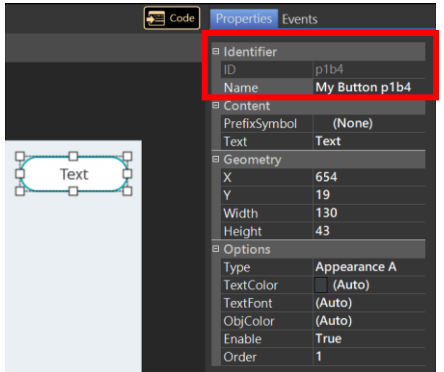
- The example HMI includes **7 buttons**: power_on, power_off, run, and move1–move4 (see right). We'll map each to code in the next section.
- Click the "**Code**" button (top-right) to generate event stubs and include statements, then paste them into your main sketch if prompted.
- After generation, you'll see the **myhmi.h** and **myhmi.cpp** represents the LVGL86 library's helper and executer.

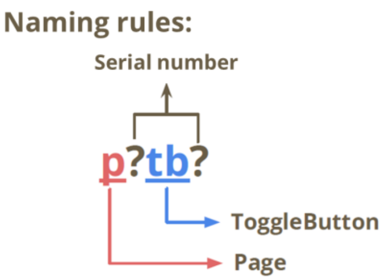
86HMI generation files



Kinematics_test HMI example

86HMI ID Naming rule



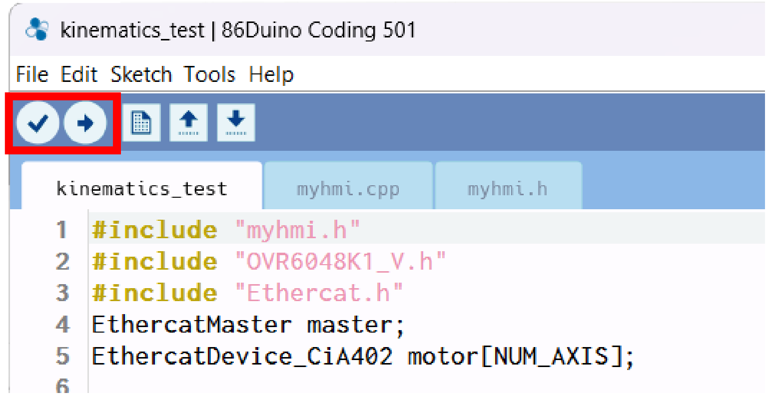


* Additional Information: In 86HMI, every object has its unique ID, and the naming rules are Page number with class serial number, as shown in the right image.

Example: Kinematics Test - 4

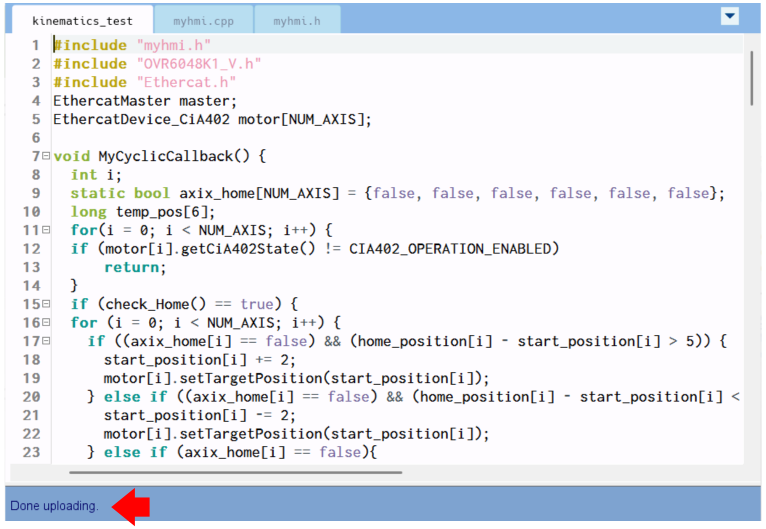
Step 4 — Build & Upload

Once the code is written, click on the toolbar to compile, and to confirm that the compilation is complete and error free, you can click to upload.



86Duino Upload & Verify icon

After uploading, you can see “**Done uploading**” in the window below.

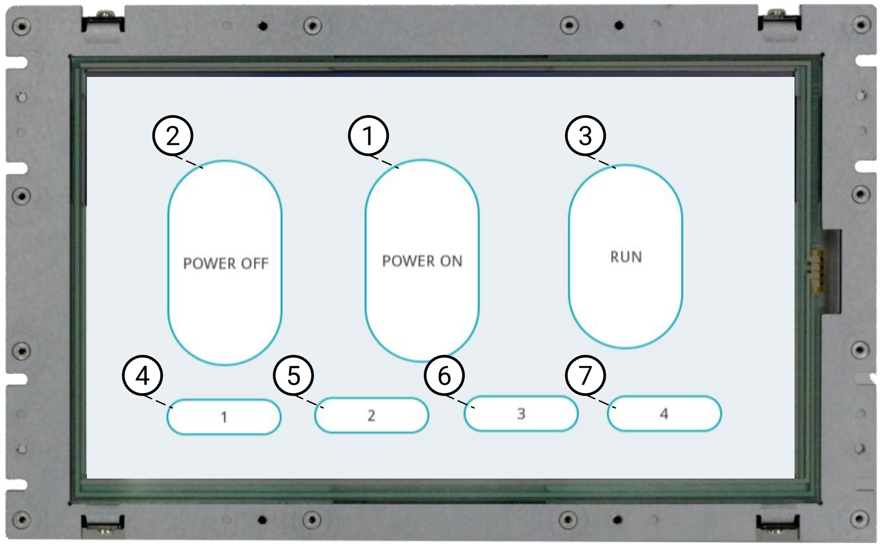


86Duino Down uploading

Example: Kinematics Test - 5

Conclusion

After you have successfully uploaded the program to the QEC-M-090T, you can view the HMI as displayed on your QEC-M-090T.



QEC-M-090T with kinematics_test example

Functions of each button

① **POWER ON (p1b1)**
`power_on();`

② **POWER OFF (p1b2)**
`power_off();`

③ **RUN (p1b3)**
`run_demo3(5.02);`

④ **1 (p1b4)**
```
Set_Des_Position(0.0, 24.0, 26.8, 0.0, -45.0, 0.0);
prepare_move();
```

⑤ **2 (p1b5)**
```
Set_Des_Position(0.0, 24.0, 26.8, 0.0, -20.0, 0.0);
prepare_move();
```

⑥ **3 (p1b6)**
```
Set_Des_Position(0.0, 24.0, 26.8, 0.0, 20.0, 0.0);
prepare_move();
```

⑦ **4 (p1b7)**
```
Set_Des_Position(0.0, 24.0, 26.8, 0.0, 45.0, 0.0);
prepare_move();
```

* Next page: We'll explain what each function does and how the motion is executed.

Usage flow & API Overview - 1

Usage flow

- **setup** → `begin()` ‣ `attach()` ‣ `setDc()` ‣ `setCiA402Mode()` ‣ `attachRTCyclicCallback()` ‣ `start()`
- **power_on** → `enable()` ‣ `poweron_init()` ‣ `home()`
- **move** → `Set_Des_Position()` ‣ `prepare_move()` / `prepare_arc_move()` (targets stream via callback)
- **loop** → HMI event.

API Overview

OVR6048K1_V.h

API	Where	Purpose	Typical use
`OVR6048K1_V_begin()`	**setup**	Init internal buffers/params	Call once before EtherCAT start
`OVR6048K1_V_loop()`	**loop**	Internal housekeeping/state	Call every loop iteration
`OVR6048K1_V_callback(long* pos)`	**cyclic callback**	Produce next target (relative/abs)	If returns true, write targets to PDO
`Set_Des_Position(x,y,z,rx,ry,rz)`	**loop/HMI** before motion	Set goal pose (required before line/arc)	Then call `prepare_move()` or arc API
`prepare_move()`	**loop/HMI**	Execute **linear** move to goal	Runs via the cyclic callback
`prepare_arc_move()`	**loop/HMI**	Execute **arc** move (uses plane/center/dir)	Set plane/center/dir first
`set_arc_plane(XY/XZ/YZ)`	**loop/HMI**	Select arc plane	Pre-req for arc
`set_arc_dir(ARC_CW/ARC_CCW)`	**loop/HMI**	Arc direction	Optional if default ok
`Set_center_offset(cx,cy,cz)`	**loop/HMI**	Arc center (relative)	Used before `prepare_arc_move()`
`poweron_init()`	**power_on**	Initialize internal parameters after enable	Call once when enabling
`emergency_stop()`	**error/safety**	Immediate stop; re-init required to resume	Use for stop

Usage flow & API Overview - 2

API Overview

Ethercat.h

API	Where	Purpose	Typical use
`master.begin()`	**setup**	Scan SubDevice, enter **Pre-Op**	First EtherCAT call
`motor[i].attach(i, master)`	**setup**	Bind SubDevice i to MDevice	For each axis
`motor[i].setDc(cycle_ns)`	**setup**	Enable Distributed Clock (DC) & set cycle time	e.g., 250'000 ns (250 µs) or 1'000'000 ns
`master.attachRTCyclicCallback(cb)`	**setup**	Register real-time cyclic callback	Your PDO loop
`master.start(cycle_ns, ECAT_SYNC)`	**setup**	Configure SM/FMMU/DC; enter **OP**	Start cyclic op
`pdoWrite()/pdoRead()`	**callback**	Write/Read process data	Typically wrapped by motor APIs

Ethercat.h > CiA402

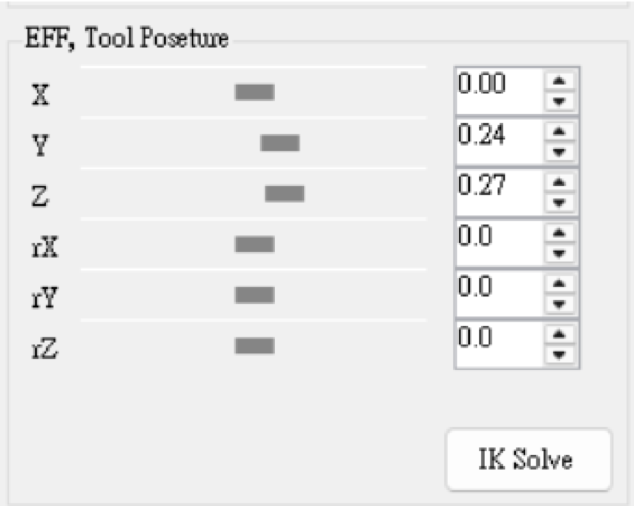
API	Where	Purpose	Typical use
`motor[i].setCiA402Mode(CIA402_CSP_MODE)`	**setup/loop**	Select CSP mode	Before start
`motor[i].enable()`	**power_on/loop**	Enter Operation Enabled	After OP & ready
`motor[i].getCiA402State()`	**callback/loop**	Query drive state	Gate motion logic
`motor[i].setTargetPosition(pulse)`	**callback/loop**	Write target (PDO)	From OVR callback/logic
`motor[i].getPositionActualValue()`	**callback/loop**	Read actual position	Homing/monitoring

Use Viewer values directly in the API

Users can take the same pose values from the **Kinematics Viewer** and apply them in code (or **ArduBlock**) to drive the arm. In the Viewer, drag the sliders or enter numbers to simulate the pose; then copy those values into the API.

The fields under **EFF, Tool Posture** — **(X, Y, Z, rX, rY, rZ)** — map **1:1** to:

```
Set_Des_Position(0.0, 24.0, 26.8, 0.0, -45.0, 0.0);
prepare_move();
```



Kinematics viewer : EFF, Tool Poseture

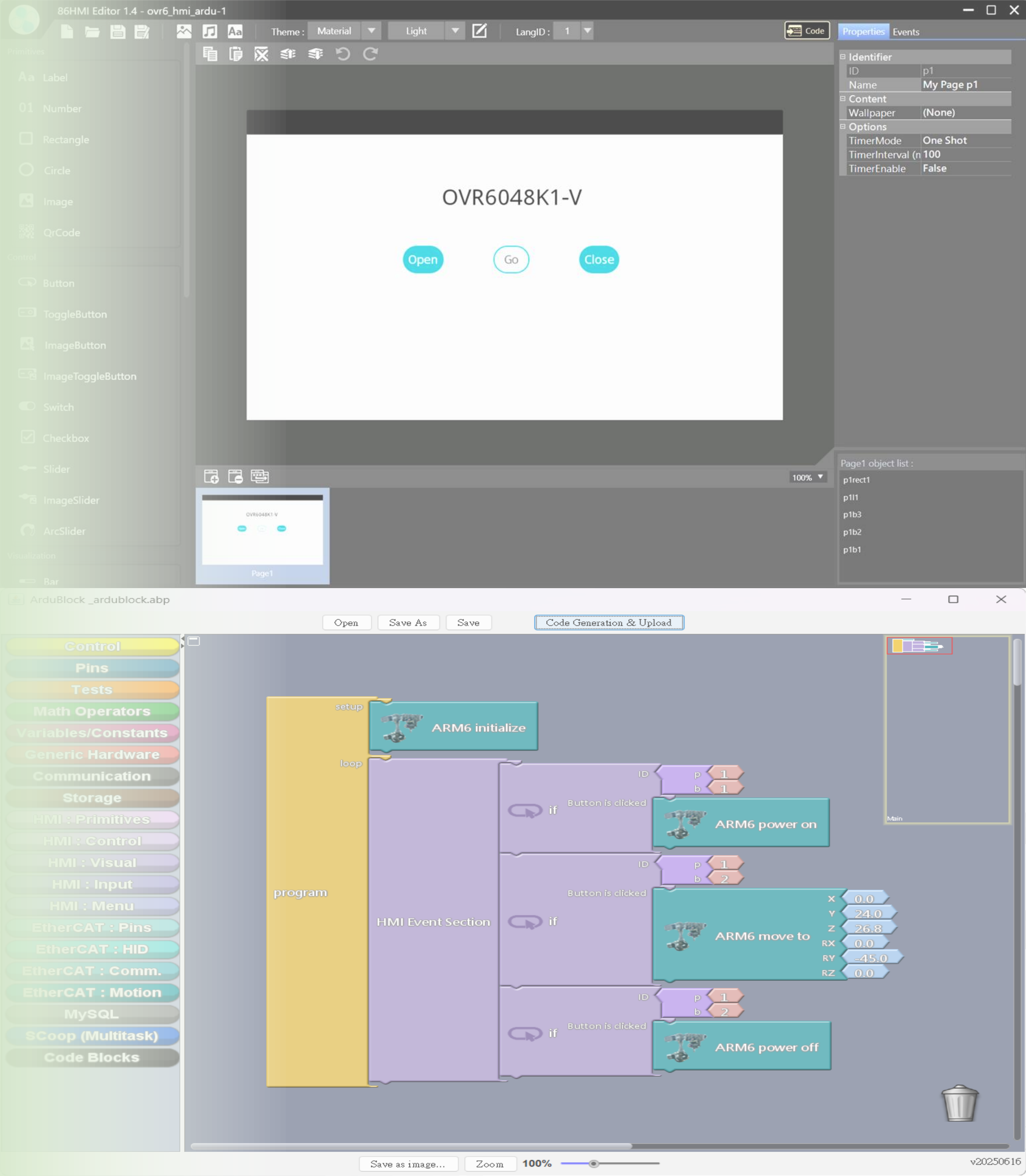
Steps:

1. In EFF, Tool Posture, set/confirm X, Y, Z, rX, rY, rZ.
2. Copy them 1:1 into `Set_Des_Position(x, y, z, rX, rY, rZ);`
3. Call `prepare_move();` to execute.

Then, you can control the arm to the same position as in the Kinematics viewer.

4. Getting Started - 1

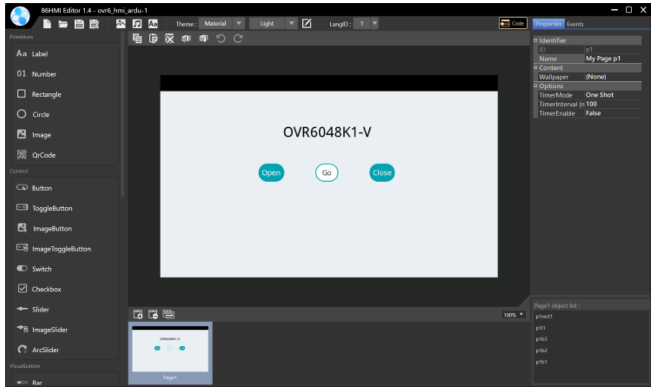
Low code Example : 86HMI + ArduBlock



86HMI + ArduBlock - Introduction 1

86HMI — UI Designer (HMI)

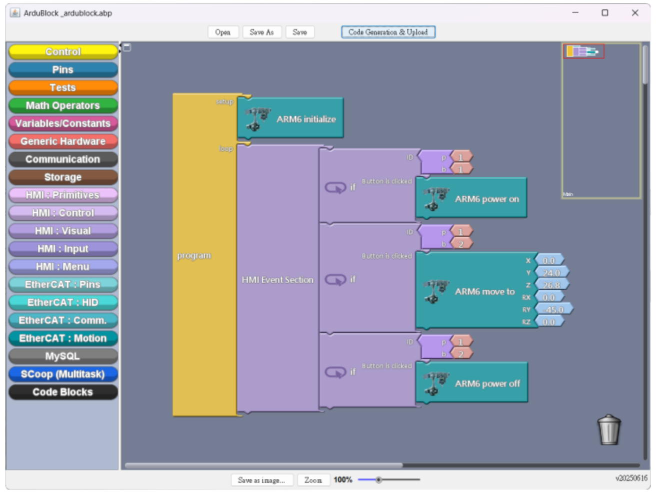
- **What it is:** A drag-and-drop screen editor based-on LVGL86 Library.
- **What you build:** Buttons, labels… bound to unique IDs (e.g., p1b1, p1b2).
- **What it generates:** C/C++ UI stubs you paste into your sketch; events are handled in an auto-generated **HMI Event Section**.
- **Why it matters:** Lets operators trigger motions (power on/off, move presets, run demo) right from the panel.



86HMI Editor

ArduBlock — Motion Logic (Blocks)

- **What it is:** A block-based editor on top of 86Duino IDE.
- **What you build:** High-level logic using blocks for **HMI**, **EtherCAT**, and **Motion**.
- **What it generates:** Valid 86Duino code that calls your APIs (e.g., `power_on()`, `Set_Des_Position()`, `prepare_move()`).



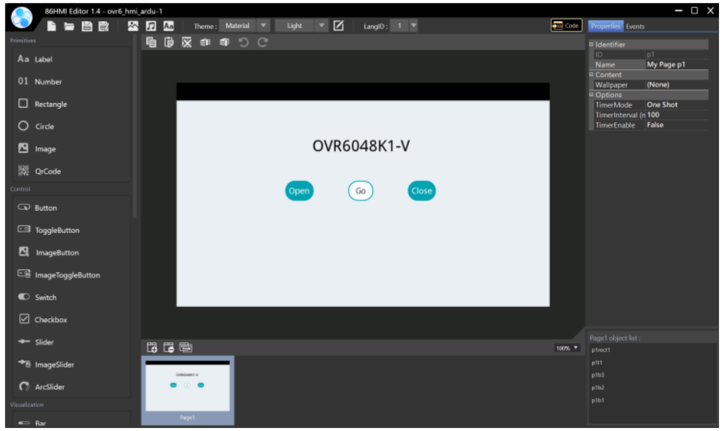
ArduBlock

*Note: the ARM6 class blocks is only for OVR6048K1-V project.

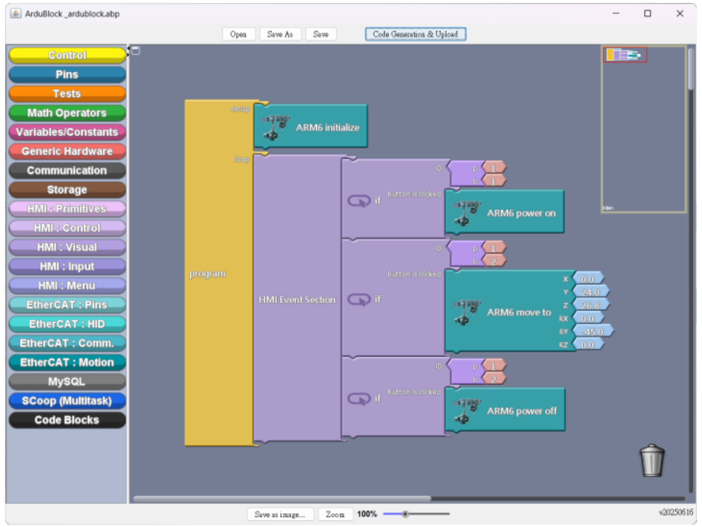
86HMI + ArduBlock - Introduction 2

Why both?

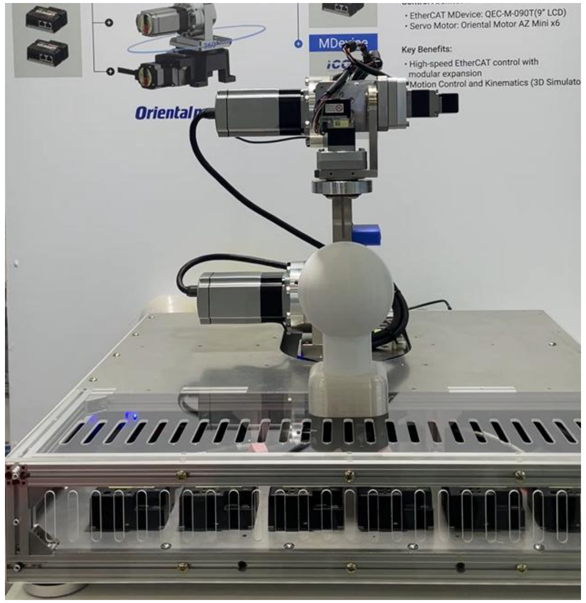
- **86HMI** designs the operator interface.
- **ArduBlock** wires those UI events to robot actions.
- **Together** they give you a low-code path from button → API call → EtherCAT motion.



86HMI Editor



ArduBlock


OVR6048K1

Example: Low code Kinematics- 1

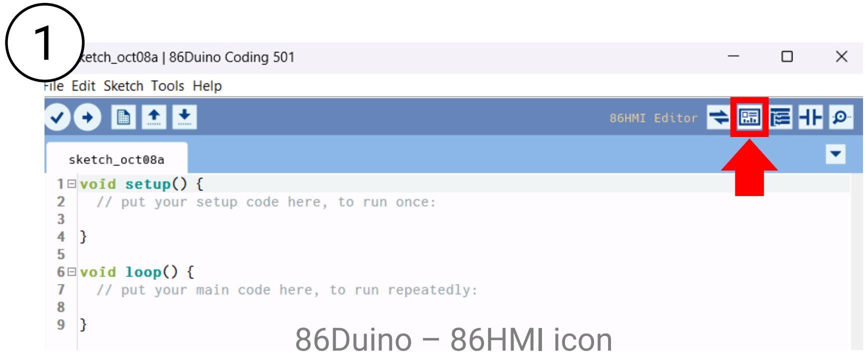
Step 1 — Open a new project and Design the HMI (86HMI)

Please refer to Example: Kinematics test for the environment setup.

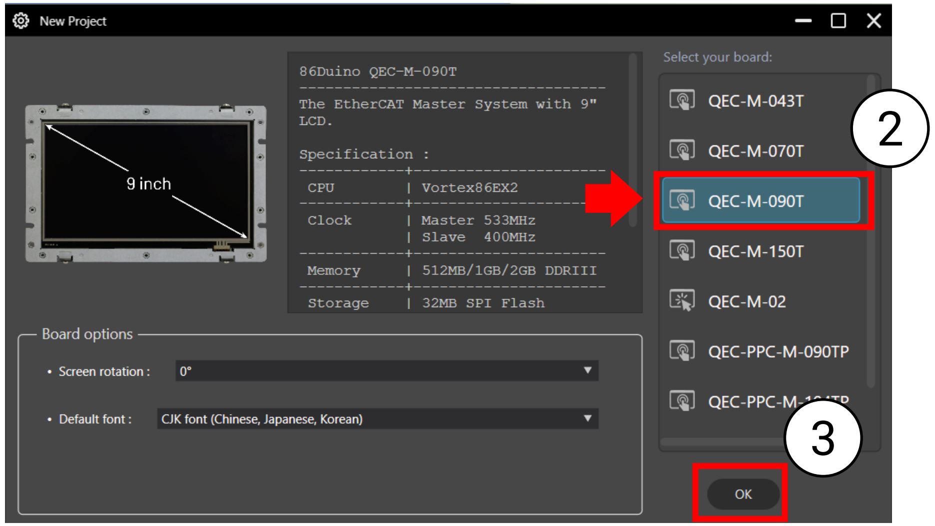
① Open 86HMI Editor.

② Choose **QEC-M-090T**.

③ Click **"OK"** button, then you can see the editor panel.



86Duino – 86HMI icon



86HMI Editor

Example: Low code Kinematics- 2

Step 2 — Create add three Buttons

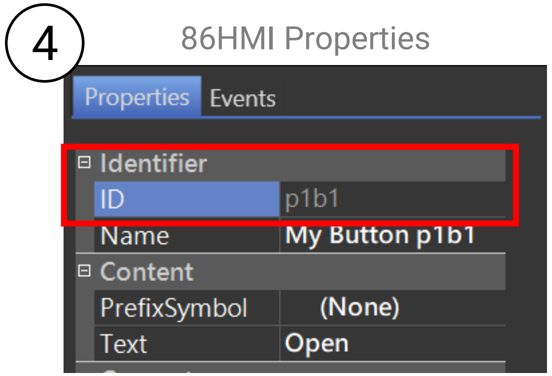
Create add three Buttons widgets: **Open**, **Go**, **Close**, on first page.

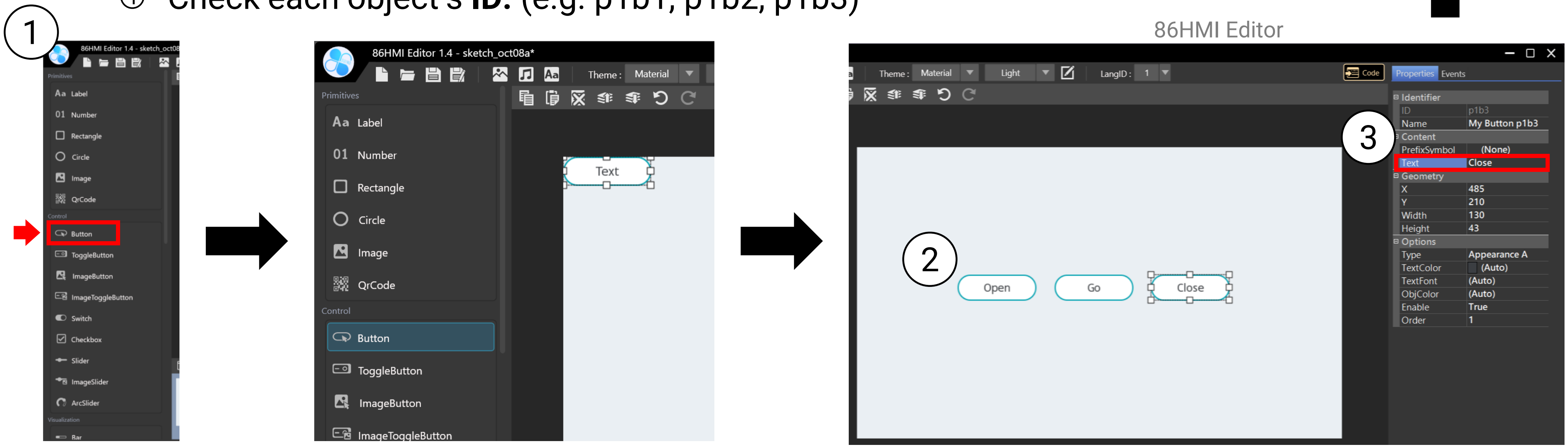
① Click the **“Button”** on the left menu.

② Duplicate three buttons.

③ Change the text to "**Open**", "**Go**", and "**Close**" in the right **properties** menu.

④ Check each object’s **ID.** (e.g. p1b1, p1b2, p1b3)

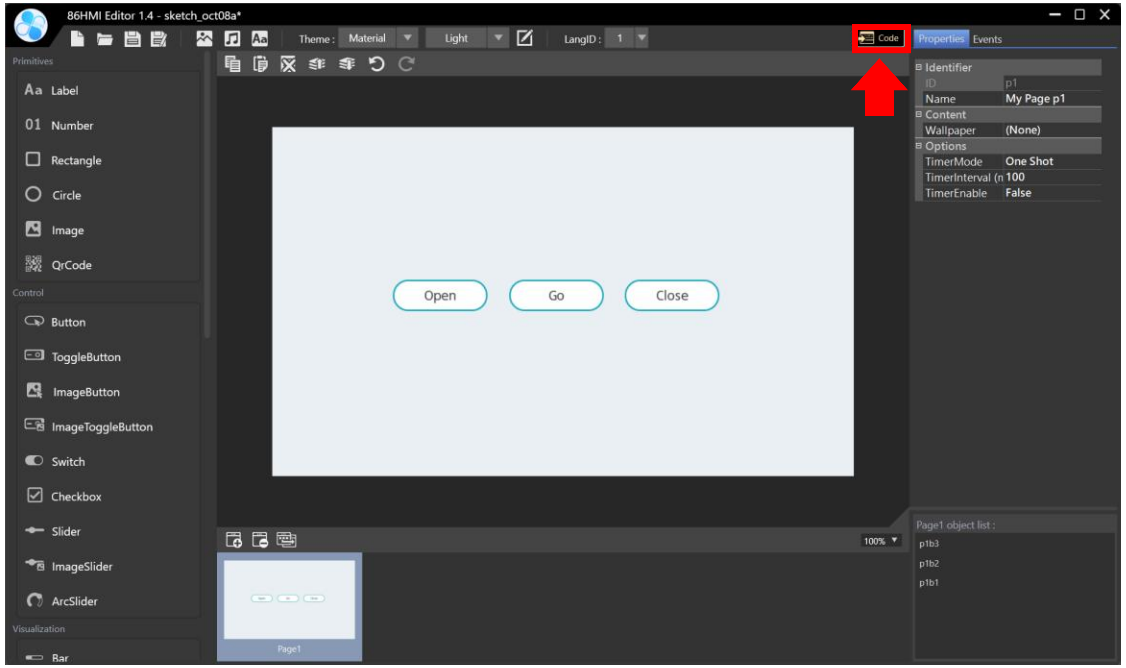




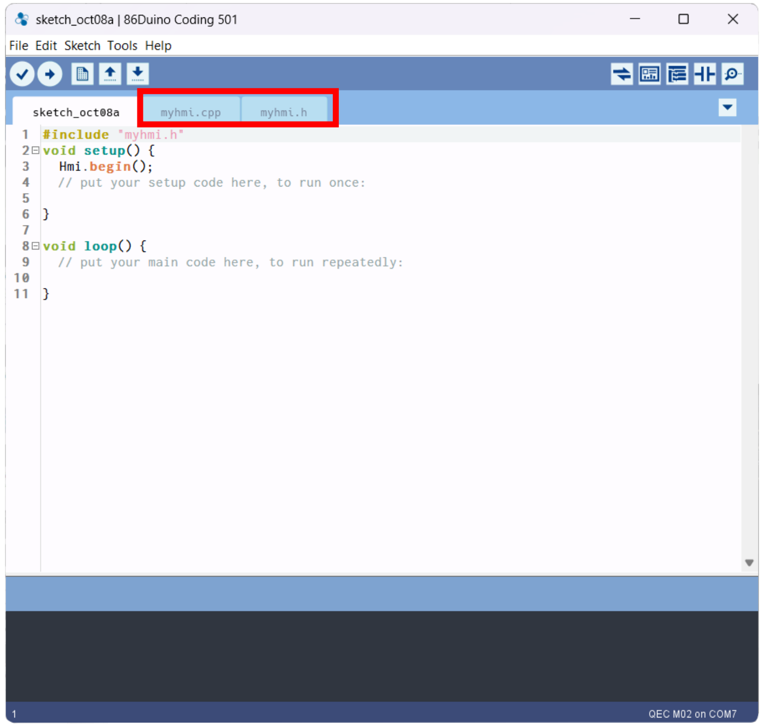
Example: Low code Kinematics- 3

Step 3 – 86HMI Code Generation

Click "**Code**" (top-right) to generate UI code and copy it into your sketch.



86HMI Editor



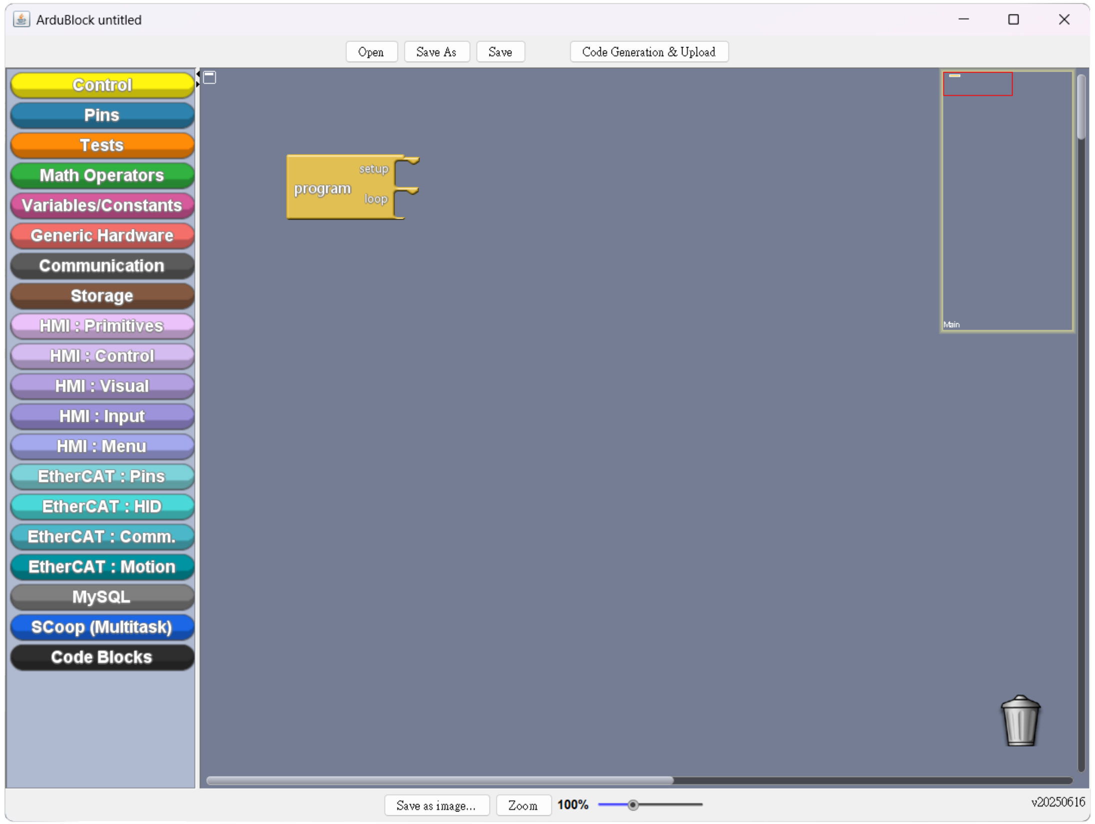
86Duino – 86HMI files

Example: Low code Kinematics- 4

Step 4 — Open the ArduBlock

Open the ArduBlock on the top-right icon.

86Duino – ArduBlock icon



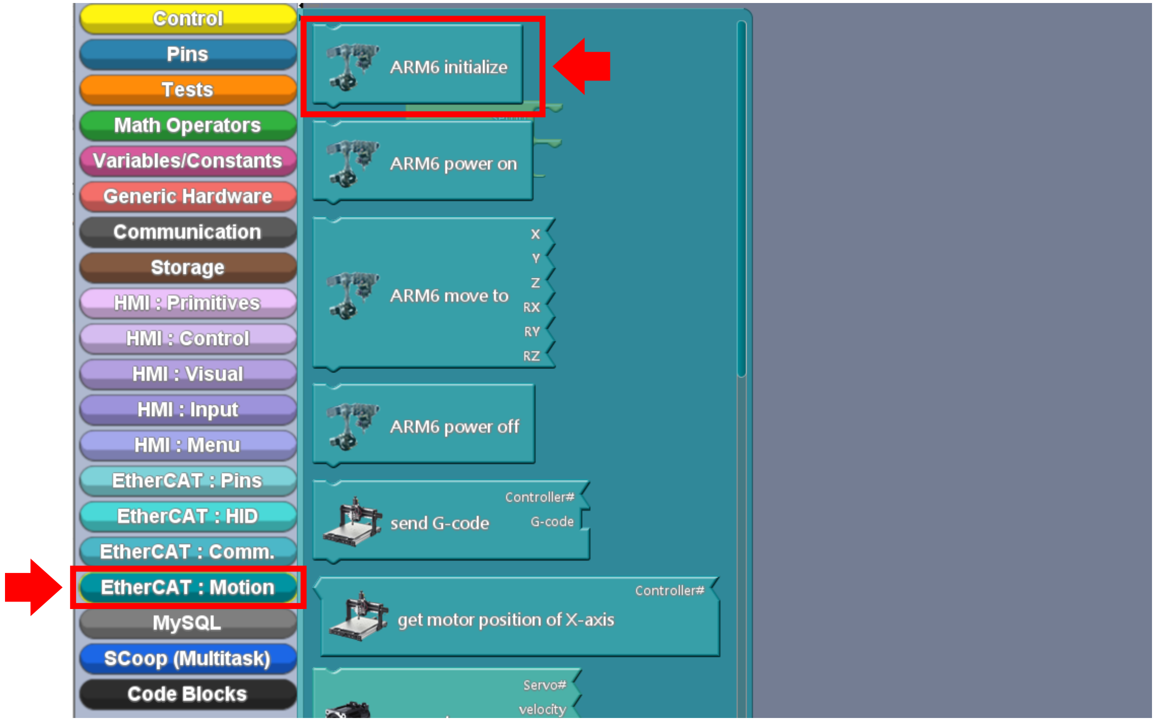
ArduBlock

Example: Low code Kinematics- 5

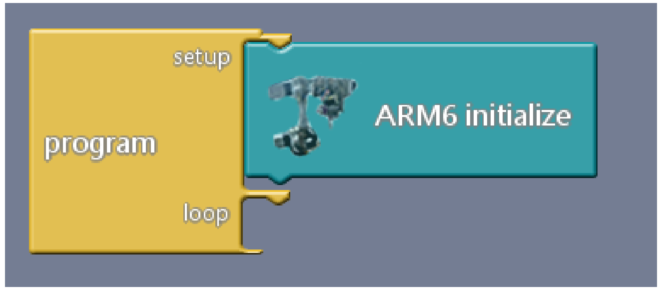
Step 5 — Put the ARM6 Initialize Block in setup

First, we need to **initialize the Orientalmotor's driver parameters**, which are already packaged in one block, called “**ARM6 initialize**”, and in the "**EtherCAT: Motion**" class.

We put it into the program's "**setup**" area.



ArduBlock – EtherCAT : Motion



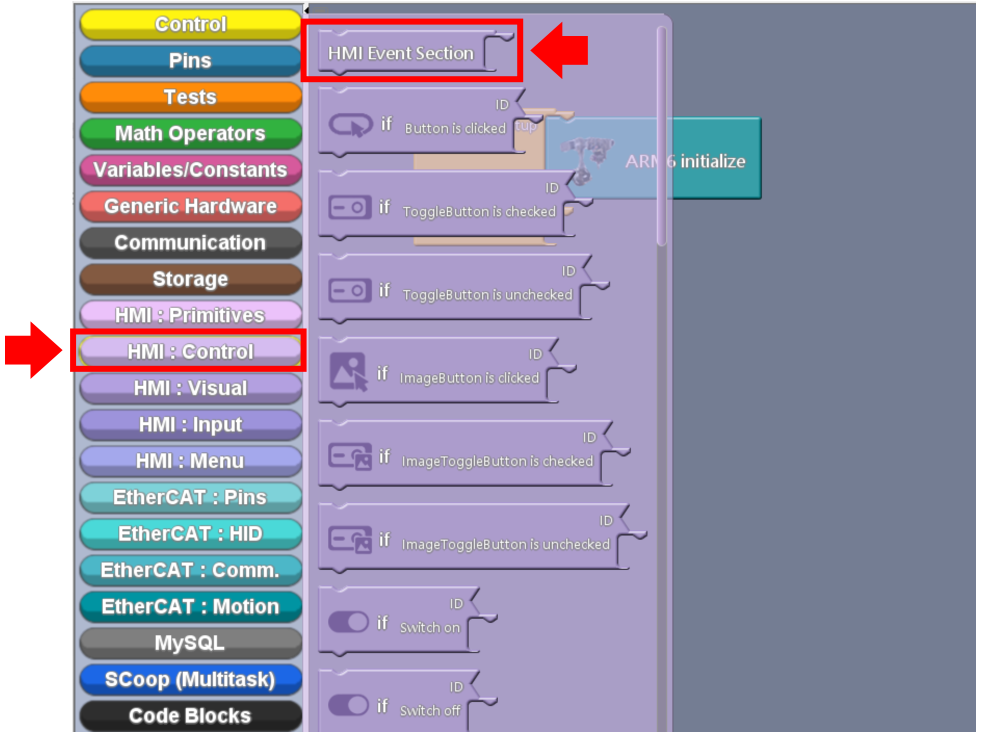
ArduBlock – ARM6 initialize

Example: Low code Kinematics- 6

Step 6 – Put the HMI event Block in the loop

Since we want to interact with the robotics arm and HMI, we have to put the HMI event in the loop.

For button events (e.g., clicked, pressing), it must be associated with the HMI event called "**HMI Event Section**", which can be found in the "**HMI : Control**" class.



ArduBlock – HMI : Control

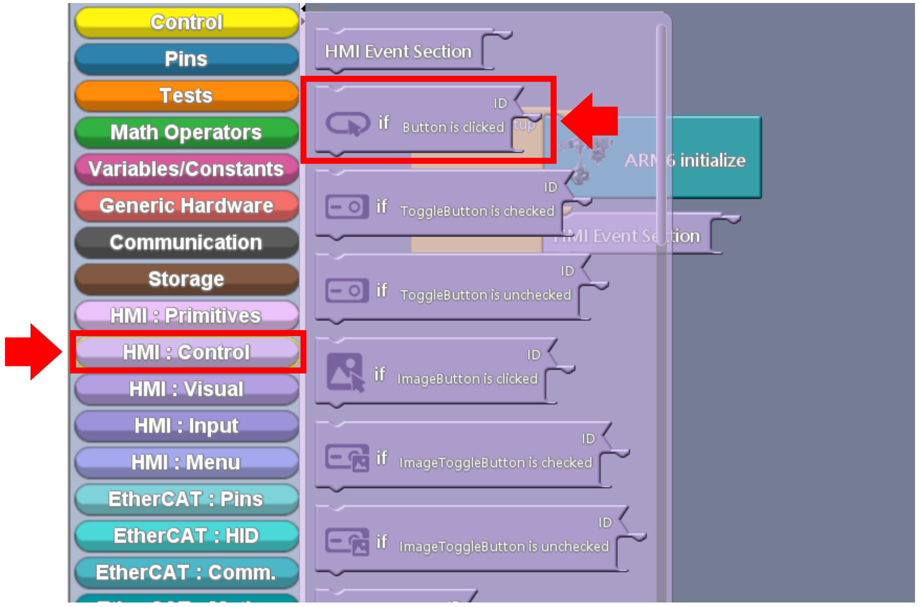
ArduBlock – HMI Event section

Example: Low code Kinematics- 7

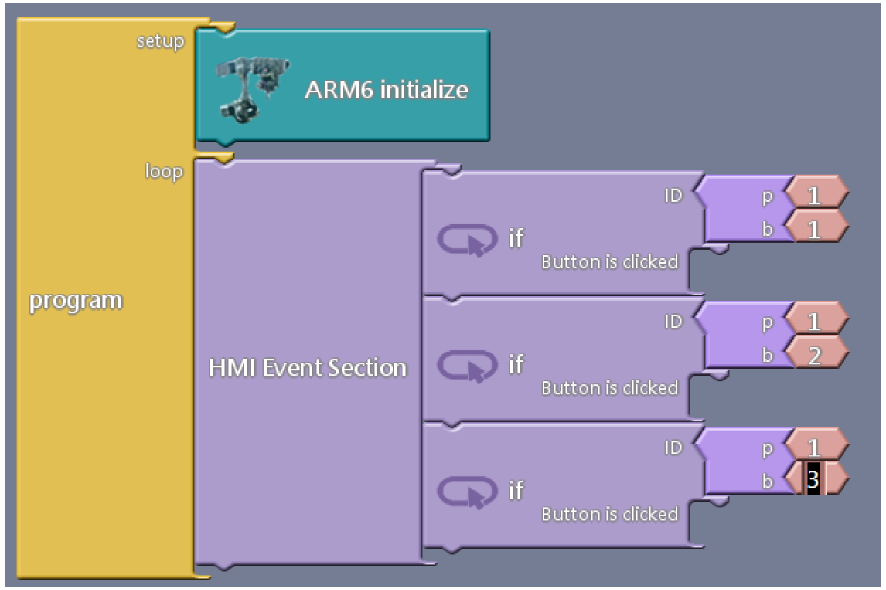
Step 7 — Put the Button event in the HMI event

Put the Button event called "**if Button is clicked**" block in the "**HMI Event Section**", which can be found in the "**HMI : Control**" class. If the specific button is clicked, the HMI event will return "**true**".

Since we need three buttons, please clone three blocks of the button clicked, and **change the ID** to **p1b1**, **p1b2**, and **p1b3**.



ArduBlock – HMI : Control



ArduBlock – if Button is clicked

Example: Low code Kinematics- 8

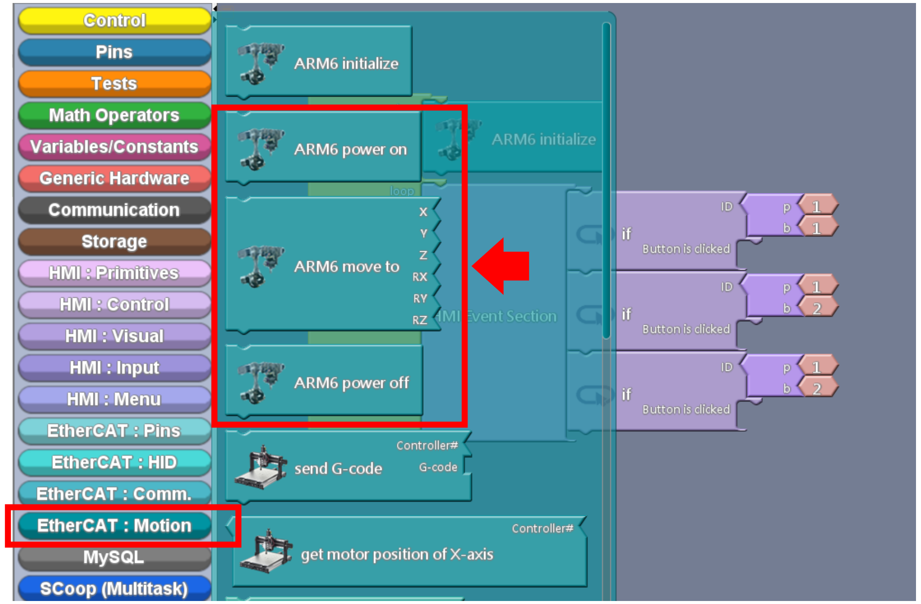
Step 8 — Put the ARM6 power-on/move-to/power-off in the Button event

The OVR6048K1's functions are already packaged into different blocks, including "**ARM6 power on**", "**ARM6 move to**", and "**ARM6 power off**", within the "**EtherCAT: Motion**" class.

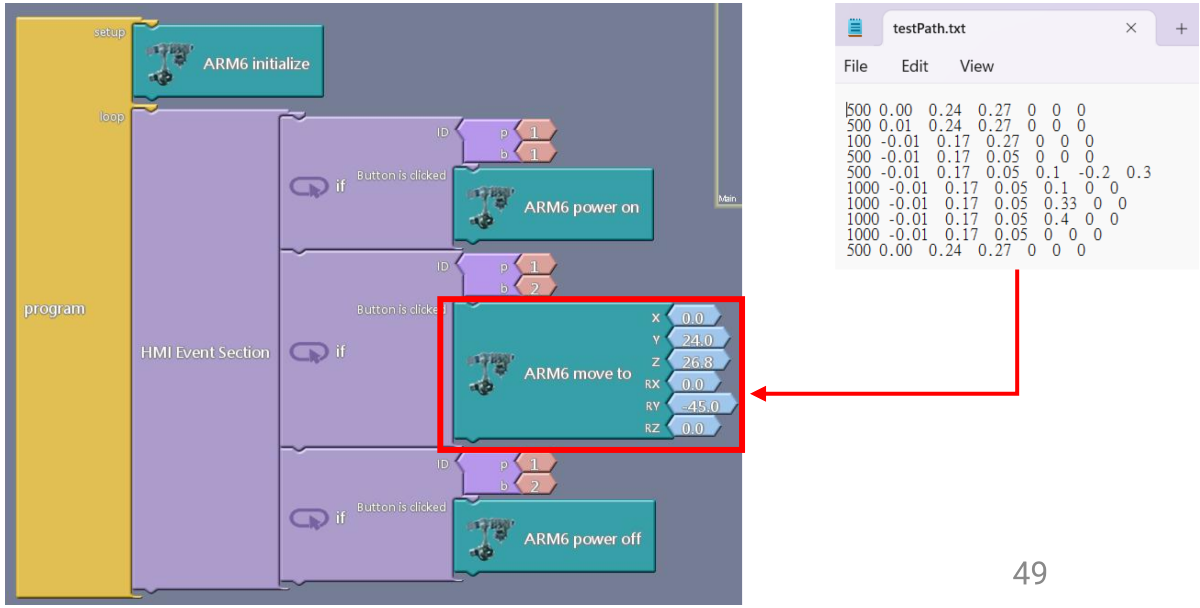
We put those OVR6048K1 function blocks in the "**if Button is clicked**" blocks as below:

- **p1b1** to "**ARM6 power on**“ / **p1b2** to "**ARM6 move to**“ / **p1b3** to "**ARM6 power off**"

*Note: Please use the **Kinematics viewer position value** for the parameter in the "**ARM6 move to**" block.



ArduBlock – EtherCAT : Motion



ArduBlock – ARM6 move to

Example: Low code Kinematics- 9

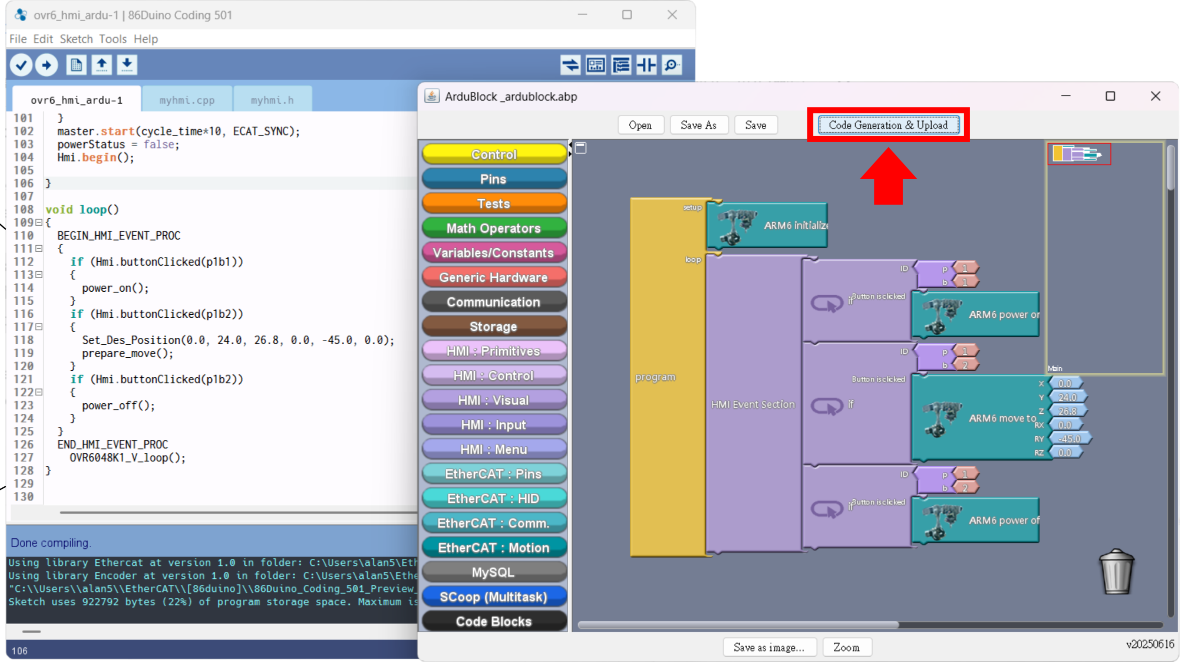
Step 9 — Code Generation & Upload

After finishing, you can click the "**Code Generate & Upload**" button to generate and upload the code to the QEC-M-090T.

This block programming establishes EtherCAT communication, configures the AZ-mini driver, and sets and controls the OVR6048K1 robotics arm using the HMI button.

```
void loop()
{
  BEGIN_HMI_EVENT_PROC
  {
    if (Hmi.buttonClicked(p1b1))
    {
      power_on();
    }
    if (Hmi.buttonClicked(p1b2))
    {
      Set_Des_Position(0.0, 24.0, 26.8, 0.0, -45.0, 0.0);
      prepare_move();
    }
    if (Hmi.buttonClicked(p1b2))
    {
      power_off();
    }
  }
  END_HMI_EVENT_PROC
    OVR6048K1_V_loop();
}
```

ArduBlock – 86Duino – HMI code

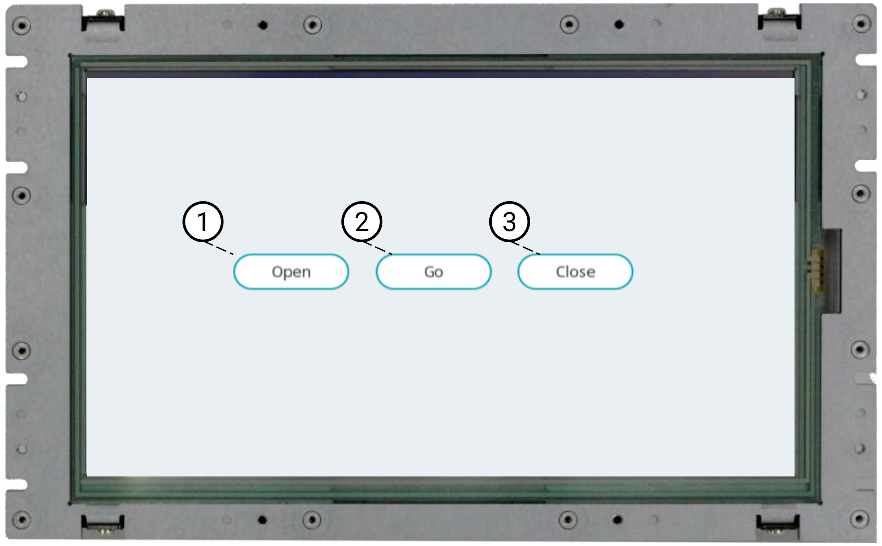


ArduBlock – 86Duino

Example: Low code Kinematics- 10

Conclusion

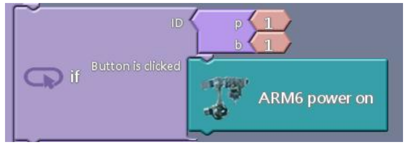
After you have successfully uploaded the program to the QEC-M-090T, you can view the HMI as displayed on your QEC-M-090T.



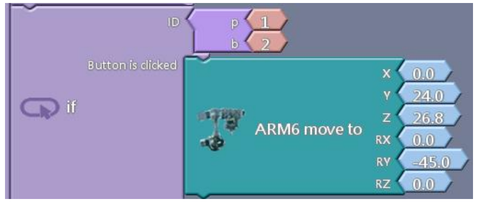
QEC-M-090T with low-code example

Functions of each button

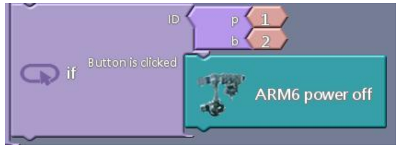
① **Open (p1b1)**



② **Go (p1b2)**

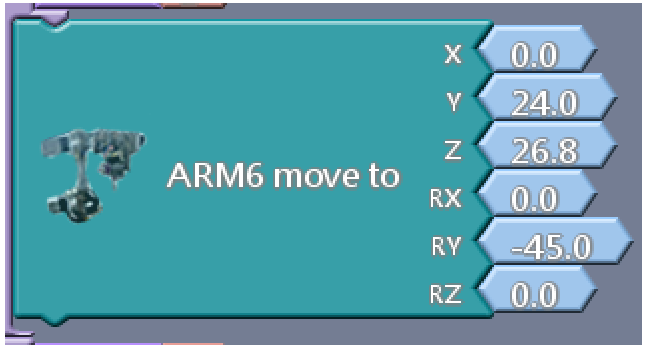


③ **Close (p1b3)**



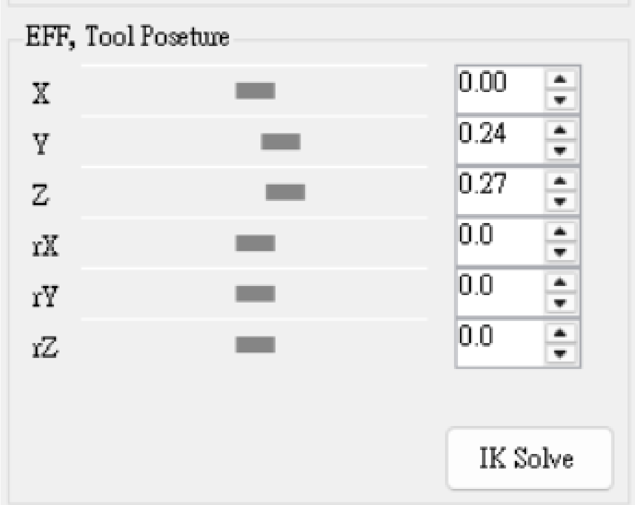
Use Viewer values directly in the ArduBlock

Users can take the same pose values from the **Kinematics Viewer** and apply them in code (or **ArduBlock**) to drive the arm. In the Viewer, drag the sliders or enter numbers to simulate the pose; then copy those values into the API.



ArduBlock Moving the Robotics Arm Block

The fields under **EFF, Tool Posture** — **(X, Y, Z, rX, rY, rZ)** — map **1:1** to:



Kinematics viewer : EFF, Tool Poseture

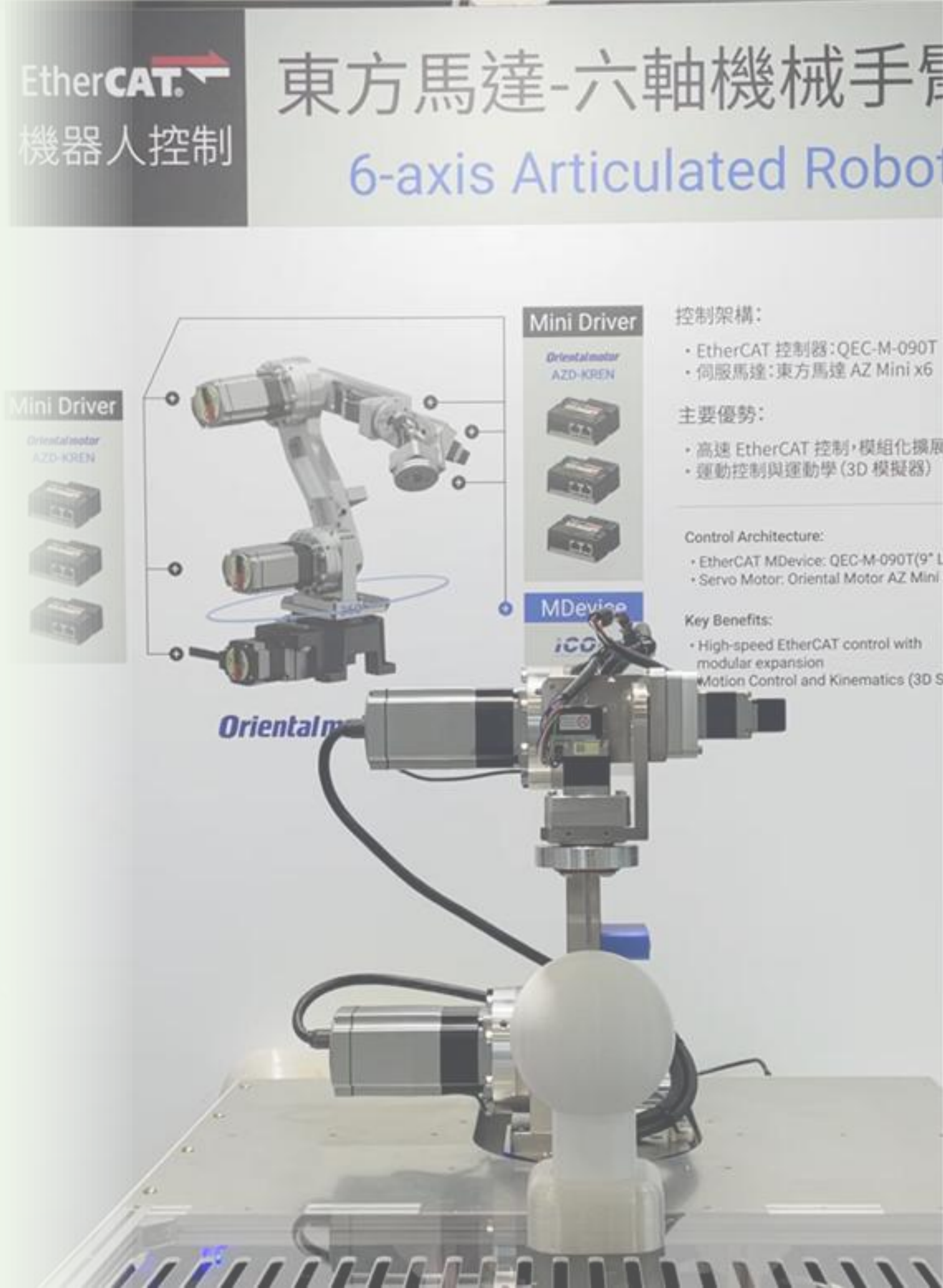
Steps:

1. In EFF, Tool Posture, set/confirm X, Y, Z, rX, rY, rZ.
2. Copy them 1:1 into `ARM6 move to(X, Y, Z, RX, RY, RZ)` Block.

Then, you can control the arm to the same position as in the Kinematics viewer.

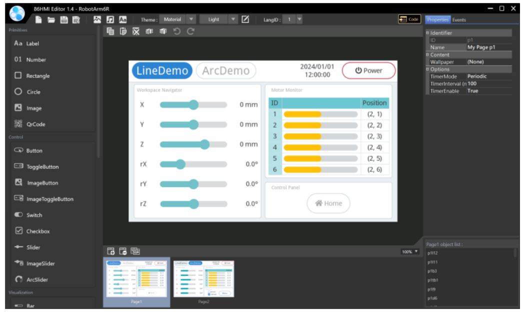
5. Getting Started - 2

Robot Arm 6R Example

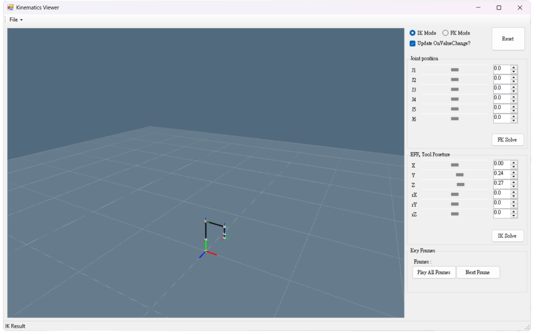


Example: RobotArm6R - Introduction

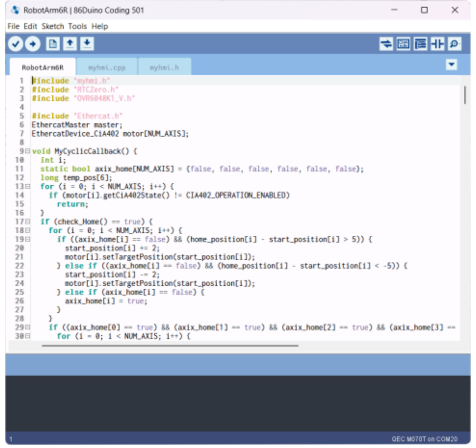
With **QEC-M-090T + AZD-KRED × OVR6048K1-V**, running **ECAT_SYNC + DC**, we execute a smooth end-effector trajectory on a **spherical surface**—from planning to simulation to one-touch HMI execution.



RobotArm6R HMI example



RobotArm6R Kinematics Viewer



RobotArm6R 86Duino example

RobotArm6R Zero Post

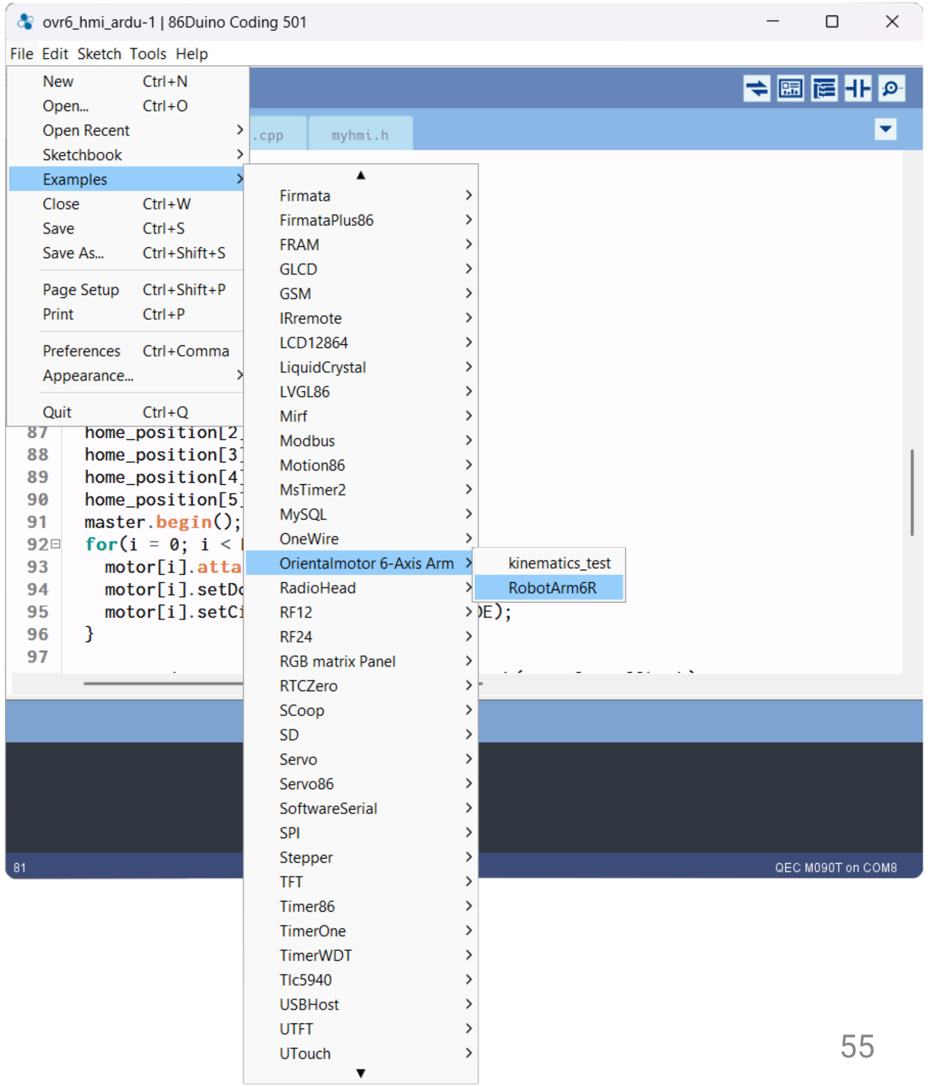
RobotArm6R example

Example: RobotArm6R - 1

Step 1 — Open the Example

- After selecting “**QEC-M090T**” under "Tools ▸ Board” and the correct Port, you can find the example set
- Open “File ▸ Examples ▸ Orientalmotor 6-Axis Arm ▸ RobotArm6R”.

86Duino Examples



Example: RobotArm6R - 2

Step 2 — View Code & Open HMI

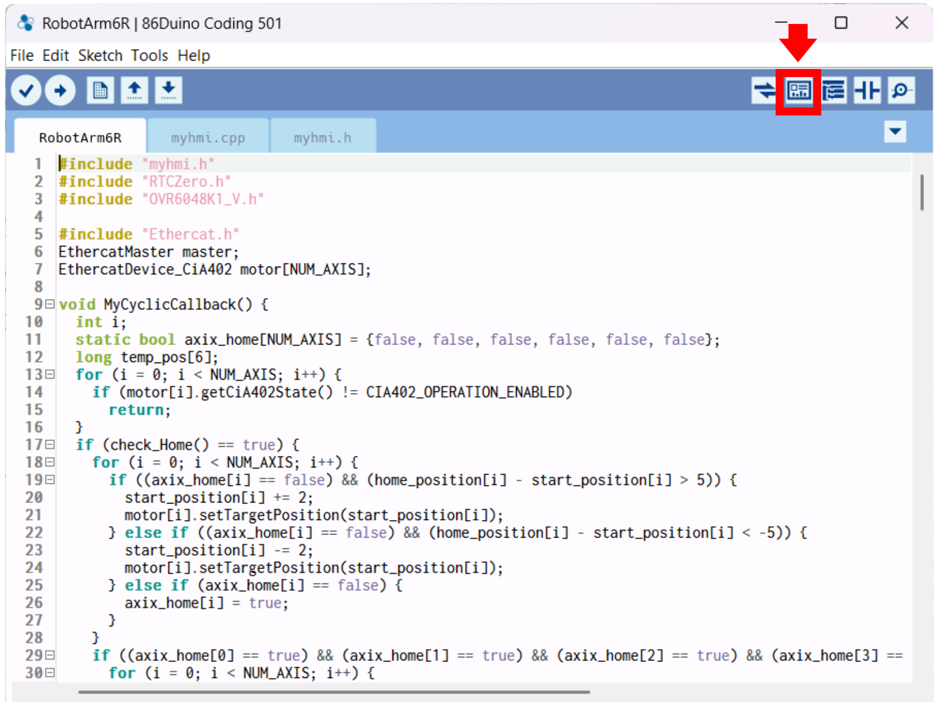
- Opening RobotArm6R loads the sketch (right). It uses three libraries:
  - **myhmi.h** — HMI helpers
  - **OVR6048K1_V.h** — robot motion helpers
  - **Ethercat.h** — EtherCAT protocol
  - **RTCZero.h** — RTC timer library
- This example uses the following settings:
  - EtherCAT Cycle Time: **250 microseconds (μs)**.
  - EtherCAT Mode: **ECAT_SYNC**.
  - Distributed Clock (DC): **Enable (250 μs).**
  - CiA-402 mode: **CSP (Cyclic Synchronous Position).**

```
master.begin();
for(i = 0; i < NUM_AXIS; i++) {
  motor[i].attach(i, master);
  motor[i].setDc(cycle_time*10);
  motor[i].setCiA402Mode(CIA402_CSP_MODE);
}

error_code = master.attachRTCyclicCallback(MyCyclicCallback);
if (error_code != 0) {
  printf("error : %d\n", error_code);
}
master.start(cycle_time*10, ECAT_SYNC); // cycle_time = 25000; // nanosecond
```

RobotArm6R.ino Code about EtherCAT

- Click the 86HMI icon (upper-right) to launch the HMI Editor.




```
#include "myhmi.h"
#include "RTCZero.h"
#include "OVR6048K1_V.h"

#include "Ethercat.h"
EthercatMaster master;
EthercatDevice_CiA402 motor[NUM_AXIS];

void MyCyclicCallback() {
  int i;
  static bool axix_home[NUM_AXIS] = {false, false, false, false, false, false};
  long temp_pos[6];
  for (i = 0; i < NUM_AXIS; i++) {
    if (motor[i].getCiA402State() != CIA402_OPERATION_ENABLED)
      return;
  }
  if (check_Home() == true) {
    for (i = 0; i < NUM_AXIS; i++) {
      if ((axix_home[i] == false) && (home_position[i] - start_position[i] > 5)) {
        start_position[i] += 2;
        motor[i].setTargetPosition(start_position[i]);
      } else if ((axix_home[i] == false) && (home_position[i] - start_position[i] < -5)) {
        start_position[i] -= 2;
        motor[i].setTargetPosition(start_position[i]);
      } else if (axix_home[i] == false) {
        axix_home[i] = true;
      }
    }
    if ((axix_home[0] == true) && (axix_home[1] == true) && (axix_home[2] == true) && (axix_home[3] ==
      for (i = 0; i < NUM_AXIS; i++) {
```

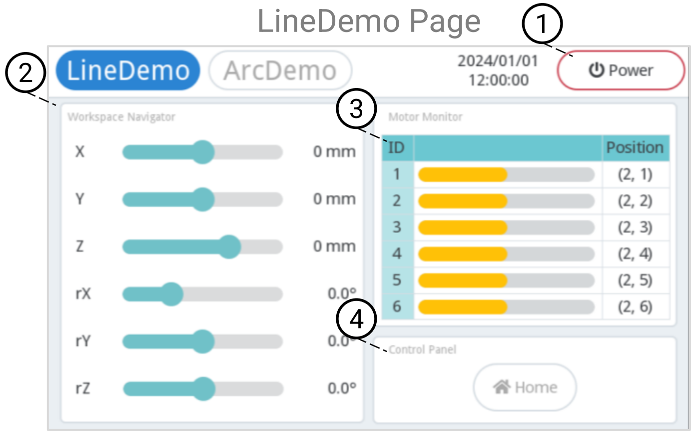
RobotArm6R.ino

Example: RobotArm6R- 3

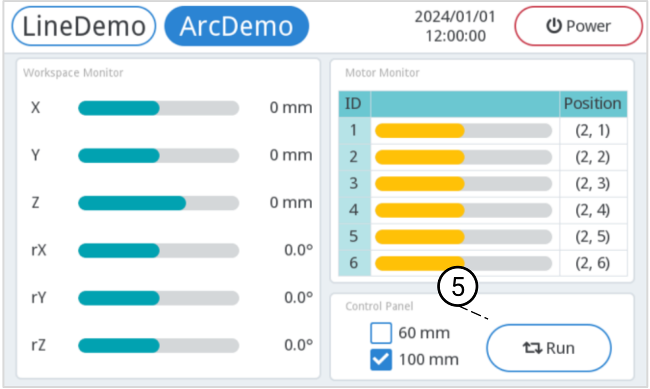
Step 3 — 86HMI Layout

- HMI includes **2 pages**: **LineDemo** and **ArcDemo**.

LineDemo Page



ArcDemo Page



Functions of each HMI objects

① **Power**
Enable or Disable all axes driver (Switch to CiA402 Enable/Disable).

② **Workspace Navigator**
Display robotics arm's workspace value, with x, y, z (mm) and rx, ry, rz (°). This area in LineDemo page can be controled, user can move the arm in the workspace, and need to after Homing and not be able to work with running demo in the same time.

③ **Motor Monitor**
Display robotics arm's position (read from motor drivers). Also show the bar with the current position.

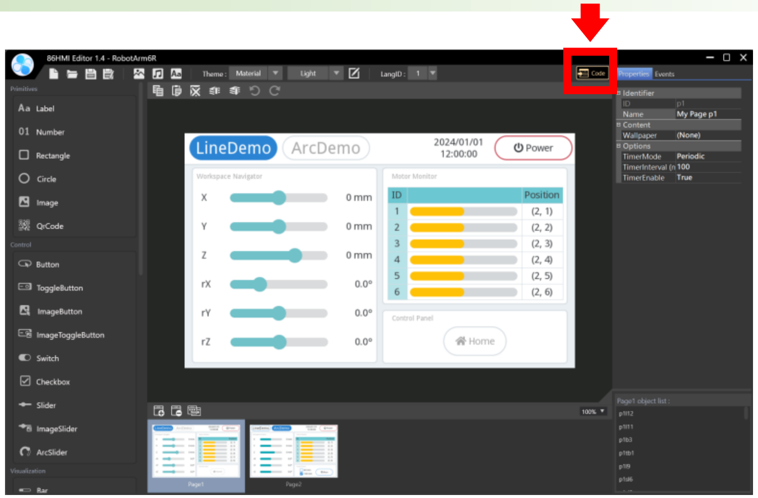
④ **Control Panel (Home)**
In LineDemo, this area with Home button to move the arm (all axes) to the home position.

⑤ **Control Panel (Run)**
In ArcDemo, this area with Run button and two radio options (60mm and 10mm). After selected the moving option, click the Run button, so robotics arm will start moving around the sphere that the size you choose; if users don't click Run button again, it'll continue the moving. Note that must be click the correct size of sphere; and do this after homing.

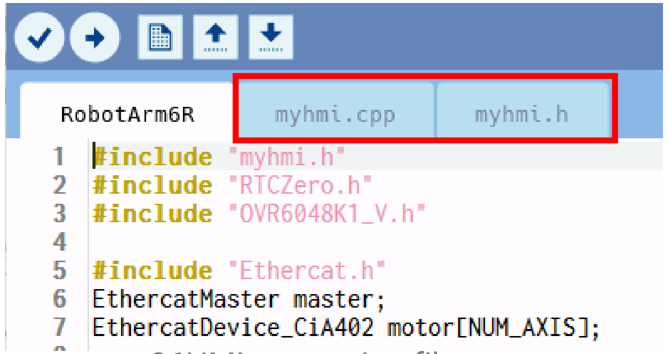
Example: RobotArm6R- 4

Step 4 — 86HMI Code Generation

- Click the “**Code**” button (top-right) to generate event stubs and include statements, then paste them into your main sketch if prompted.
- After generation, you'll see the **myhmi.h** and **myhmi.cpp** represents the LVGL86 library's helper and executer.



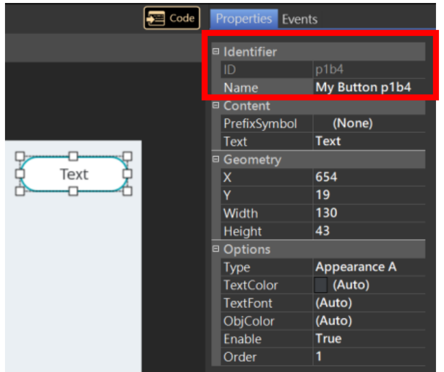
RobotArm6R HMI example

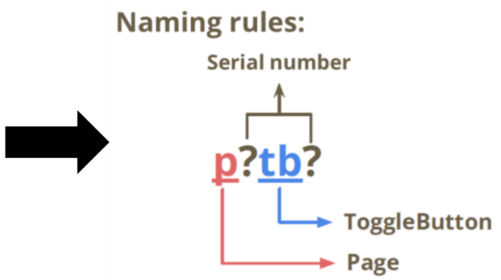


86HMI generation files

* Additional Information: In 86HMI, every object has its unique ID, and the naming rules are Page number with class serial number, as shown in the right image.

86HMI ID Naming rule

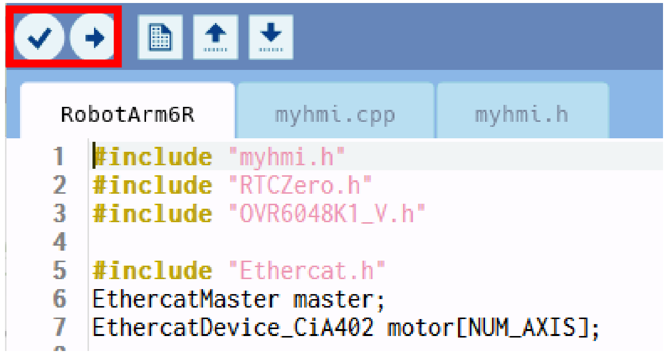




Example: RobotArm6R - 5

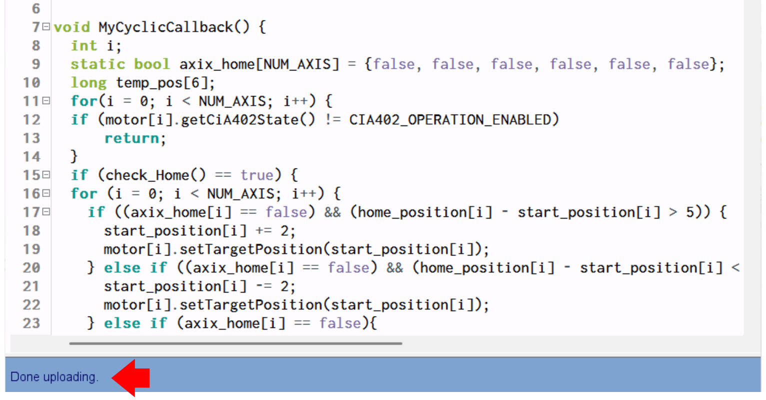
Step 5 — Build & Upload

Once the code is written, click on the toolbar to compile, and to confirm that the compilation is complete and error free, you can click to upload.



86Duino Upload & Verify icon

After uploading, you can see “**Done uploading**” in the window below.

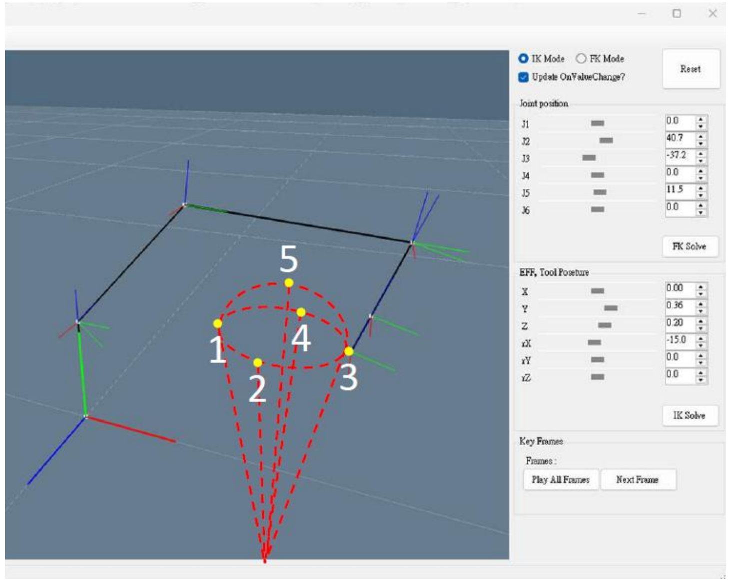


86Duino Done uploading

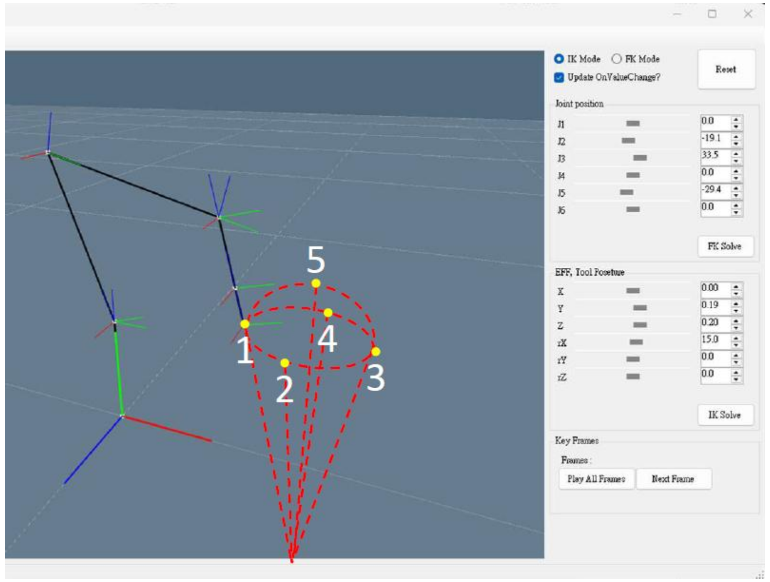
Example: RobotArm6R illustrate

In the RobotArm6R example, it'll move around a sphere of radius. There are two demos: 60mm or 100mm Sphere.

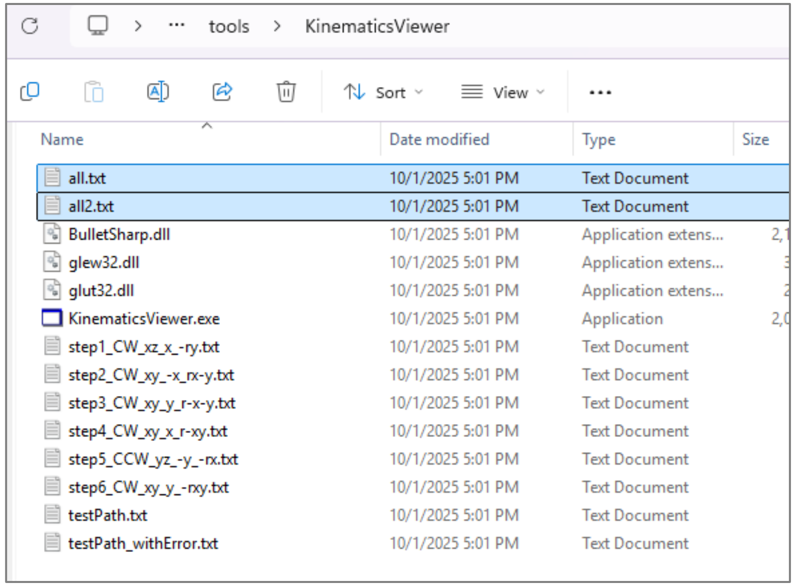
Users can view the path value in the **Kinematics Viewer**; the examples are called **all.txt** and **all2.txt**.



robotArm6R Moving in Kinematics viewer -1



robotArm6R Moving in Kinematics viewer -2



robotArm6R Sphere demo path

Example: RobotArm6R - Video

Conclusion

(Video: https://youtu.be/9-X3WKAfU1o?si=HUMSyUU-MeYdyDMh)

END

Thank you!!

For more detailed information, please get in touch with us.

QEC Website

ICOP Website

Orientalmotor Website