Content

[1. LabVIEW Demo 1](#_Toc11351)

[1.1 Environment setup 1](#_Toc1294)

[1.2 LabVIEW Demo description 4](#_Toc14031)

[1.2.1 Project description 4](#_Toc841)

[1.2.2 Program block diagram description 5](#_Toc30809)

[1.3 Common error description 7](#_Toc19110)

[1.3.1 TCP connection error 7](#_Toc15479)

[1.3.2 Error during installing 7](#_Toc30029)

[1.4 API description 8](#_Toc1693)

[1.4.1 Get PTPJumpParams 8](#_Toc11045)

[1.4.2 Set and get joint point location parameter（Set/Get PTPJointParams） 8](#_Toc20561)

[1.4.3 Set and get coordinate point location parameter（Set/Get PTPCoordinateParams） 9](#_Toc5107)

[1.4.4 Set and get joint coordinate system JOG parameter（Set/Get JOGJointParams） 9](#_Toc11602)

[1.4.5 Set and get Cartesian coordinate system JOG parameter（Set/Get JOGCoordinateParams） 10](#_Toc31573)

[1.4.6 Set arm orientation（SetArmOrientation） 10](#_Toc11799)

[1.4.7 Get real-time pose（GetPose） 11](#_Toc15835)

[1.4.8 Calibrate robotic arm（ResetPose） 11](#_Toc23330)

[1.4.9 Get alarm state（GetAlarmsState） 11](#_Toc3970)

[1.4.10 Set/Get user coordinate（Set/Get UserCoordinate） 11](#_Toc14325)

[1.4.11 Set/Get tool coordinate system（Set/Get ToolCoordinate） 12](#_Toc28504)

# LabVIEW Demo

This document aims to help users to get familiar with Dobot M1 common APIs and quickly build a development environment.

## Environment setup

This demo is based on LabVIEW, so LabVIEW needs to be installed.

This article takes the Windows 10 operating system as an example for installation and configuration instructions, please replace it according to the actual situation.

**Steps:**

**Step 1** Install LabVIEW, Take "LabVIEW 2018" as an example, the installation method will not be explained in detail.

**Step 2** Install LabVIEW plug-in vipm-17.0.2007-windows-setup.exe. The installation method will not be explained in detail.

**Step 3** Double click installation pacakge . This installation pacakage is based on calling SDK packaged by DOBOT API in LabVIEW environment. It can be installed through the VI Package Manager. （For labview17, please install )

**Step 4** Click on “Install” button and start installation. LabVIEW software will be opened automatically during installation process. Do not close LabVIEW software before installation is complete. As shown in Figure 1.1 below:

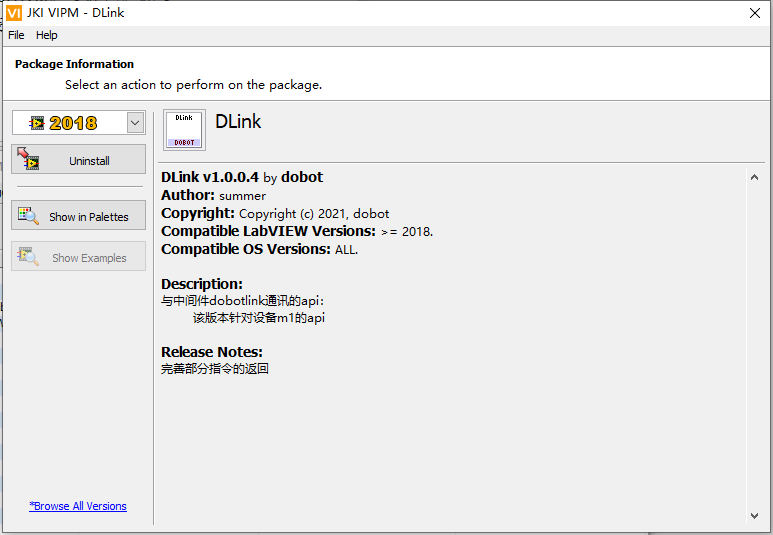


Fig 1.1 install SDK

**Step 5** After installation is successful, you can select the "Installed" mode on the main page to see that the Dlink installation package has been installed. As shown in Figure 1.2 below.

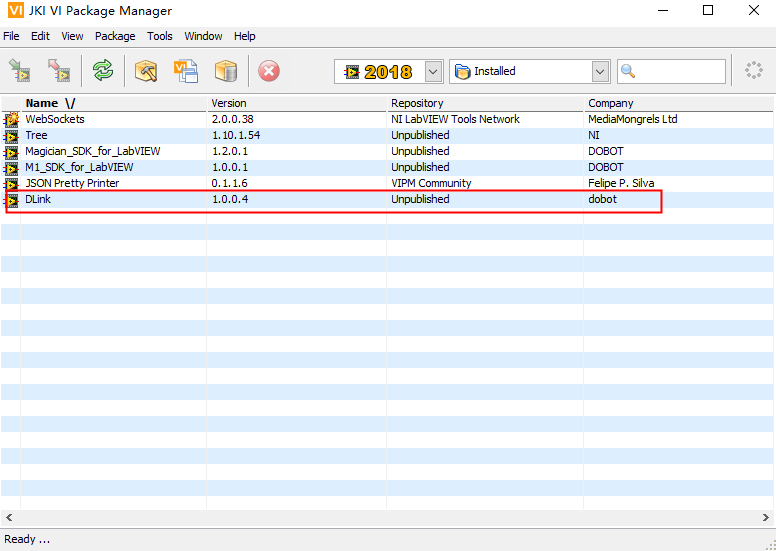
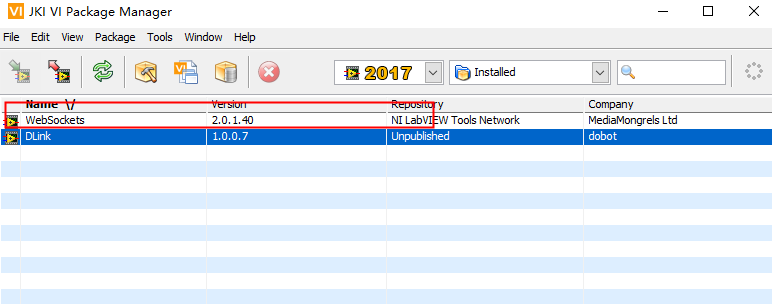


Fig 1.2 Install sdk suceeded

**Step 6** Click the link (vipm://mediamongrels\_ltd\_lib\_websockets\_api), and it will pop up a window as below to install “websocket” plugin.



It will be as below after installed “websocket” plug-in successfully:



**Step 7** Start LabVIEW and import LabVIEW Demo, as shown in Figure 1.3.

"template.vi" provides users with a template for how to call the dobotlink api.

"Alarm.vi" is the front panel for calling the alarm command.

"Arc.vi" is the front panel for calling circular interpolation commands.

"coordinate.vi" is the front panel for calling coordinate system transformation commands.

"Cp.vi" is the front panel for calling continuous motion trajectory commands. "device.vi" is the front panel for calling device information commands.

"Io.vi" is the front panel for calling the port input and output commands.

"jog.vi" is the front panel for calling the jog function.

"Ptp.vi" is the front panel for calling PTP commands.

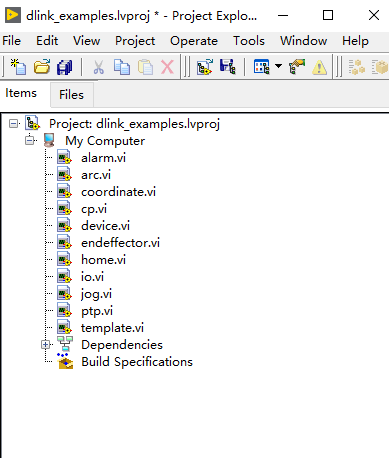


Fig 1.3 Open LabVIEW Demo

**Step 8** Open “device.vi” in LabVIEW Demo project. It will show the front panel of “device.vi”, “Ctrl+E” open block diagram, as shown in Figure 1.4.

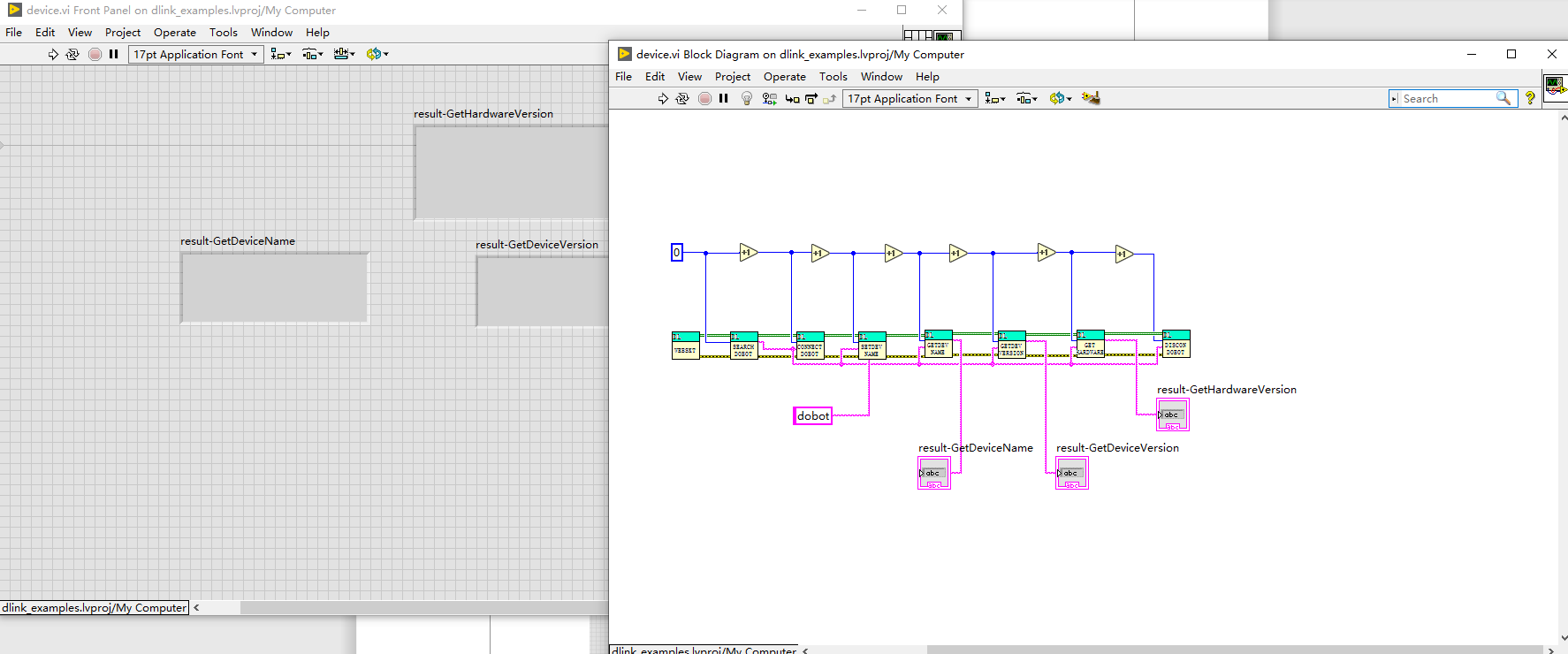


Fig. 1.4 device front panel and block diagram

**Step 9** Connect with robot arm, single click to run.

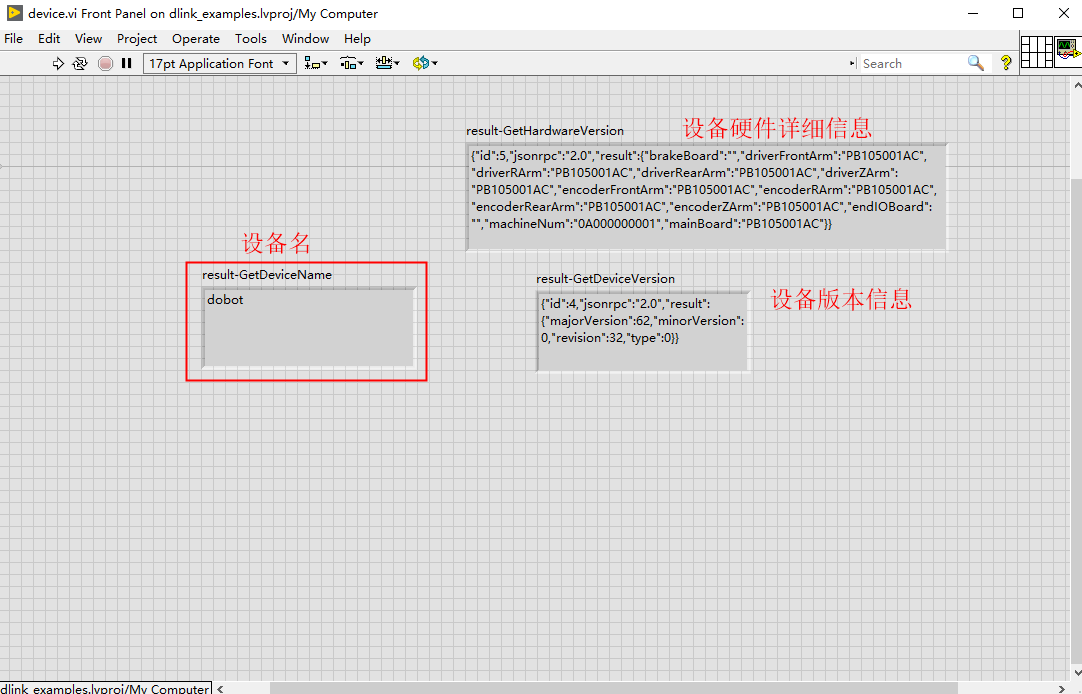


Fig 1.5 Results of device.vi

## LabVIEW Demo description

### Project description

Open “template.vi”, front panel and block diagram is as shown in Figure 1.6.

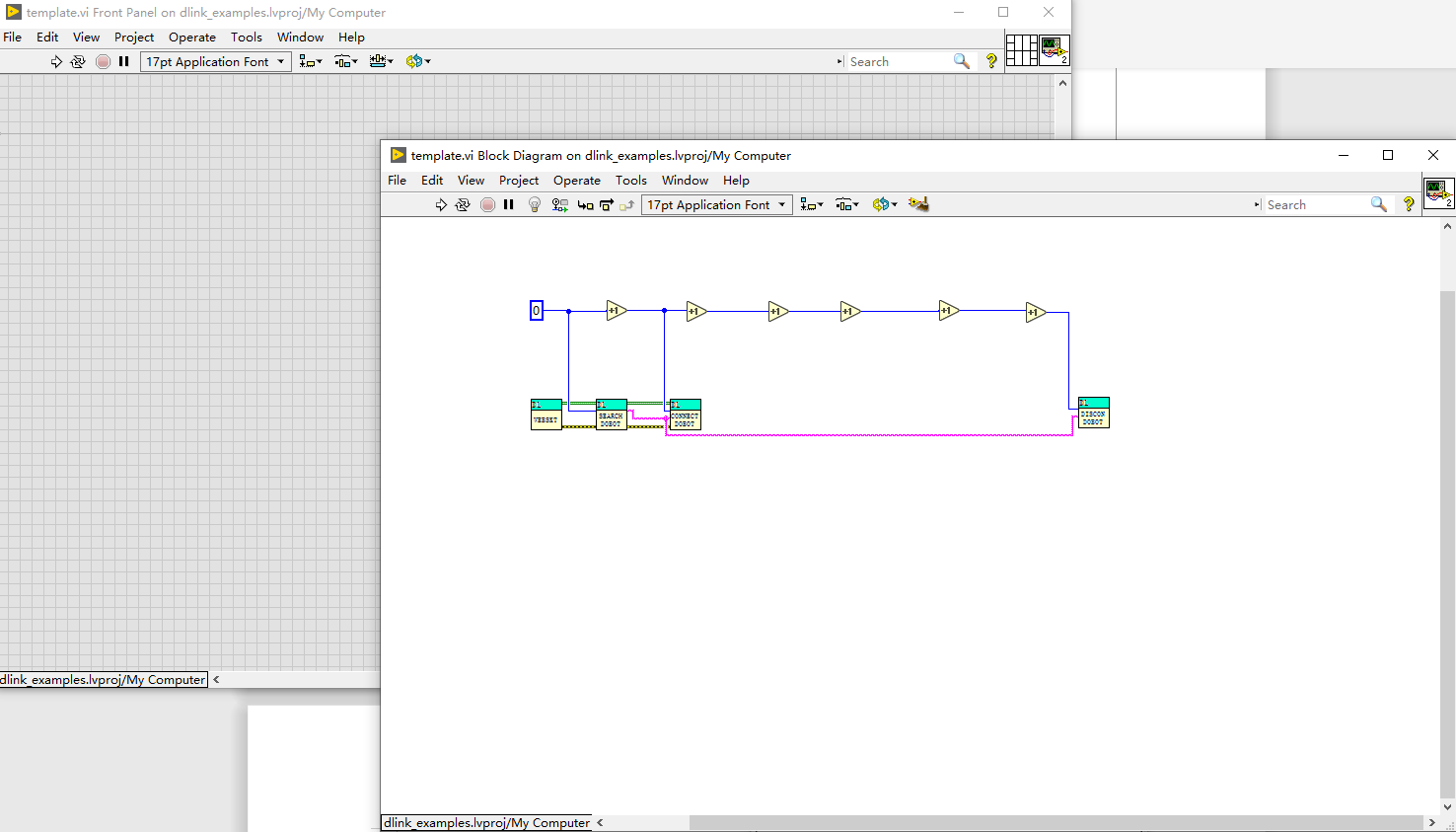


Fig 1.6 template front panel and block diagram

In the program block diagram age, users could right click to select “user library>DLink” to call DobotLink API as shown in Fig 1.7. Please refer to <Dobot M1 API description> if you need more information for API.

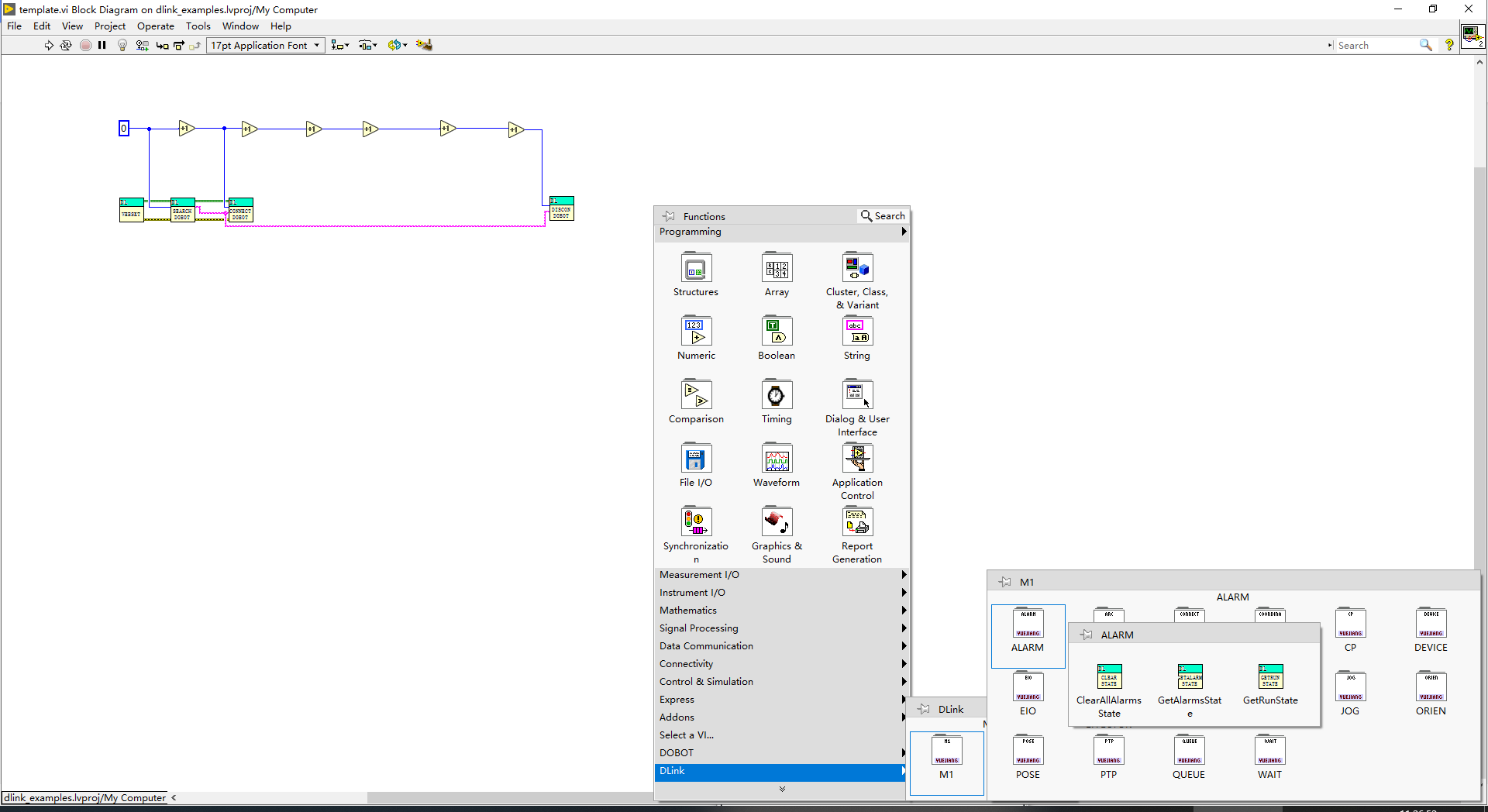
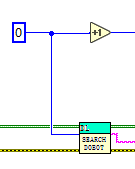


Fig 1.7 Call API

### Program block diagram description

This demo is based on communication between dobotlink and websocket. So you need to call the websocket api . Device commands basic connection ports are as follows: (id: each instruction has an exclusive ID, cannot be repeated, websocket input/output, error input/output, result output).

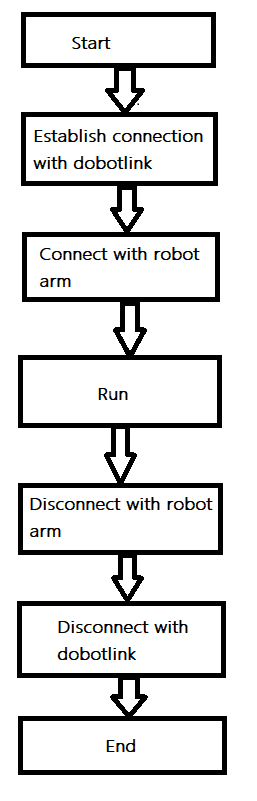


Fig 1.8 Demo flow chart

Description of each block diagram of the Demo:

* + - * 1. Establish connection with dobotlink

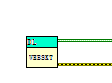


Fig 1.9 Establish connection with dobotlink

* + - * 1. Connect robot arm

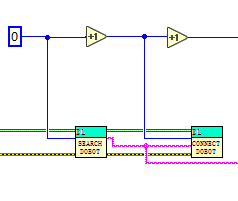


Fig 1.10 Connect robot arm

* + - * 1. Disconnect robot arm

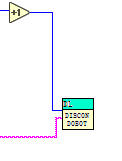


Fig 1.11 Execute PTP instructions by the instruction queue

1. Disconnect with dobotlink (after program finishes run, it will disconnect with dobotlink automatically, it’s ok to skip this one)



Fig 1.12 disconnect

## Common error description

### TCP connection error

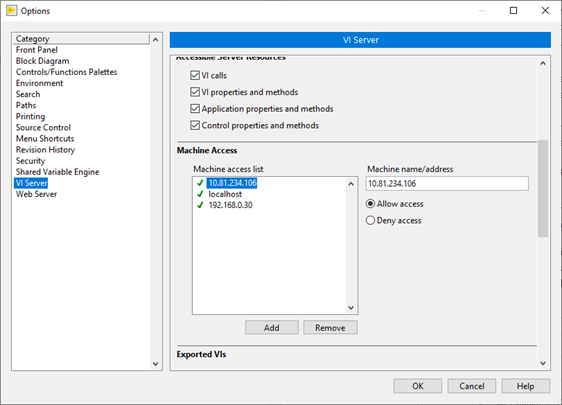
#### lALPBFDk9hkfzzDNAh7NBTk_1337_542

Solution: Please confirm whether dobolink is started normally, an icon  will be displayed in the lower right corner is dobotlink is successfully started.

### Error during installing

#### Please refer to the following solution:

LabVIEW -> Options -> VI server -> Machine Access -> add -> "localhost"



Please try to install again after setting above.

## API description

For all API descriptions, please check the document dobot-m1-api.pdf, below are some API descriptions that are different from document dobot-m1-api.pdf.

### Get PTPJumpParams

isUsingZLimit as output, usually is default as true, no need to consider it.

### 

### Set and get joint point location parameter（Set/Get PTPJointParams）

These two commands are used to set and get the reproduction speed parameters, including the joint reproduction speed and acceleration. The speed-related parameters set by this command are only applicable to the reproduction movement and are invalid for the JOG function.

Max. Speed of J1-J4: 180, 180, 1000, 1000

Max. Acceleration of J1-J4: 20000, 20000, 80000, 50000

#### Set joint coordinate system point location parameter(SetPTPJointParams), is used to control the speed of reproduced movement.

Table 1 Set joint point location parameter

|  |  |
| --- | --- |
| Command | **SetPTPJointParams** |
| Function | Set joint point location parameter |
| Parameter | velocity: array(float):speed setting of J1-J4 respectively  acceleration: array(float):acceleration setting of J1-J4 respectively |
| Return | Whether set successfully, return to true or false |

Table 2 get joint point location parameter

|  |  |
| --- | --- |
| Command | **GetPTPJointParams** |
| Function | get joint point location parameter |
| Parameter | None |
| Return | velocity: array(float):speed setting of J1-J4 respectively  acceleration: array(float):acceleration setting of J1-J4 respectively |

### 

### Set and get coordinate point location parameter（Set/Get PTPCoordinateParams）

Table 3 Set Cartesian coordinate system point location parameter

|  |  |
| --- | --- |
| Command | **SetPTPCoordinateParams** |
| Function | Set Cartesian coordinate system point location parameter |
| Parameter | xyzVelocity：float //xyz axis speed in PTP mode  rVelocity：float //End-effector speed in PTP mode  xyzAcceleration：float //xyz axis acceleration in PTP mode  rAccleration：float //End-effector acceleration in PTP mode |
| Return | Whether set successfully, return to true or false |

Table 4 Get Cartesian coordinate system point location parameter

|  |  |
| --- | --- |
| Command | **GetPTPCoordinateParams** |
| Function | Get Cartesian coordinate system point location parameter |
| Parameter | None |
| Return | xyzVelocity: float //xyz axis speed in PTP mode  rVelocity: float //End-effector speed in PTP mode  xyzAcceleration: float //xyz axis acceleration in PTP mode  rAccleration: float //End-effector acceleration in PTP mode |

### Set and get joint coordinate system JOG parameter（Set/Get JOGJointParams）

Table 5 Set joint point location parameter

|  |  |
| --- | --- |
| Command | **SetJOGJointParams** |
| Function | Set joint coordinate system point location parameter |
| Parameter | velocity: array(float) // Speed of J1-J4  acceleration: array(float) //Acceleration of J1-J4 |
| Return | Whether set successfully, return to true or false |

Table 6 Get joint point location parameter

|  |  |
| --- | --- |
| Command | **GetJOGJointParams** |
| Function | Get joint coordinate system point location parameter |
| Parameter | None |
| Return | velocity: array(float) //Speed of J1-J4  acceleration: array(float) //Acceleration of J1-J4 |

### Set and get Cartesian coordinate system JOG parameter（Set/Get JOGCoordinateParams）

Table 7 Set coordinate axis point location parameter

|  |  |
| --- | --- |
| Command | **SetJOGCoordinateParams** |
| Function | Set coordinate axis point location parameter |
| Parameter | velocity: array(float) //This command sets parameters of the Cartesian coordinate system, namely speed of X, Y, Z and R axes.  acceleration: array(float) //This command sets parameters of the Cartesian coordinate system, namely acceleration of X, Y, Z and R axes. |
| Return | Whether set successfully, return to true or false |

Table 8 Get coordinate axis point location parameter

|  |  |
| --- | --- |
| Command | **GetJOGCoordinateParams** |
| Function | Get coordinate axis point location parameter |
| Parameter | None |
| Return | velocity: array(float) //This command sets parameters of the Cartesian coordinate system, namely speed of X, Y, Z and R axes.  acceleration: array(float) //This command sets parameters of the Cartesian coordinate system, namely acceleration of X, Y, Z and R axes. |

### Set arm orientation（SetArmOrientation）

Table 9 Set arm orientation

|  |  |
| --- | --- |
| Command | **SetArmOrientation** |
| Function | Set arm orientation |
| Parameter | orientation: int (0:lefty 1: righty) |
| Return | Whether set successfully, return to true or false |

### Get real-time pose（GetPose）

Table 10 Get real-time pose

|  |  |
| --- | --- |
| Command | **GetPose** |
| Function | Get real-time pose |
| Parameter | None |
| Return | Xyzr: array(float) //Robotic arm coordinate system: x, y, z, r  jointAngle: array(float) //Angle of robotic arm joint(J1, J2, J3, J4) |

### Calibrate robotic arm（ResetPose）

Table 11 Calibrate robotic arm

|  |  |
| --- | --- |
| Command | **ResetPose** |
| Function | Calibrate robotic arm |
| Parameter | frontAngle1: float // frontAngle1 and frontAngle2 are the angle of the forearm when the robot arm reaches the same point with the left and right hands respectively  frontAngle2: float |
| Return | Whether set successfully, return to true or false |

### Get alarm state（GetAlarmsState）

Table 12 Get alarm state

|  |  |
| --- | --- |
| Command | **GetAlarmsState** |
| Function | Get alarm state |
| Parameter | None |
| Return | state: array(int) //Each byte in the array “state” can identify the alarm state of 8 alarm items, and the MSB is in the high bit and the LSB is in the low bit. |

### Set/Get user coordinate（Set/Get UserCoordinate）

Table 13 Set user coordinate

|  |  |
| --- | --- |
| Command | **SetUserCoordinate** |
| Function | Set user coordinate |
| Parameter | xyzr: array(float) |
| Return | Whether set successfully, return to true or false |

Table 14 Get user coordinate

|  |  |
| --- | --- |
| Command | **GetUserCoordinate** |
| Function | Get user coordinate |
| Parameter | None |
| Return | xyzr: array(float) |

### Set/Get tool coordinate system（Set/Get ToolCoordinate）

Table 15 Set tool coordinate system

|  |  |
| --- | --- |
| Command | **SetToolCoordinate** |
| Function | Set tool coordinate system |
| Parameter | xyzr: array(float) |
| Return | Whether set successfully, return to true or false |

Table 16 Get tool coordinate system

|  |  |
| --- | --- |
| Command | **GetToolCoordinate** |
| Function | Get tool coordinate system |
| Parameter | None |
| Return | xyzr: array(float) |