



# **RealMan Robot rm\_ros\_interface User Manual**

## **V1.0**



---

**RealMan Intelligent Technology (Beijing) Co., Ltd.**



### Revision History:

No.	Date	Comment
V1.0	11/20/2023	Draft



# Content

1. rm_ros_interface Package Description .....	4
2. rm_ros_interface Package Use .....	4
3. rm_ros_interface Package Architecture Description .....	4
3.1 Overview of Package Files .....	4
4. rm_ros_interface message description .....	5
1. Joint error code: Jointerrorcode.msg .....	5
2. Clearing the joint's error code: Jointerrclear.msg .....	5
3. All coordinate system names: Getallframe.msg .....	6
4. Joine motion: Movej.msg .....	6
5. Linear motion: Movel.msg .....	6
6. Circular motion: Movec.msg .....	6
7. Joint space planning to target pose: Movejp.msg .....	7
8. Joint transmission: Jointpos.msg .....	7
9. Pose transmission: Cartepos.msg .....	8
10. Current robotic arm state (Angle + Euler angle): Armoriginalstate.msg ...	8
11. Current arm state (radians + quaternion): Armstate.msg .....	9
12. Getting the software version: Armsoftversion.msg .....	9
13. Gripper's pick: Gripperpick.msg .....	10
14. Gripper's pick (grripper's pick-on): Gripperpick.msg .....	10
15. Gripper reaching the given position: Gripperset.msg .....	10
16. Force-position mixing control: Setforceposition.msg .....	11
17. Six-axis force data: Sixforce.msg .....	11
18. Setting the dexterous hand posture: Hand posture.msg .....	12
19. Setting the dexterous hand action sequence: Handseq.msg .....	12
20. Setting the angles of various degrees of freedom for the dexterous hand: Handangle.msg .....	13
21. Setting the dexterous hand action sequence: Handspeed.msg .....	13
22. Setting the force threshold for the dexterous hand: Handforce.msg .....	13



23.	Transmissive force-position mixing control compensation (angle):	
	Forcepositionmovejoint.msg .....	13
24.	Transmissive force-position mixing control compensation (pose):	
	Forcepositionmovejoint.msg .....	14
25.	Speed open loop control (lifting mechanism): Liftspeed.msg .....	15
26.	Position closed-loop control (lifting mechanism): Lift height.msg .....	15
27.	Getting the state of the lifting mechanism: Liftstate.msg .....	16
29.	Getting the state of the lifting mechanism: Liftstate.msg .....	16
30.	Getting the state of the lifting mechanism: Liftstate.msg .....	17



## 1. rm\_ros\_interface Package Description

The main function of the rm\_ros\_interface package is to provide necessary message files for the robotic arm to run under the framework of ROS2. In the following text, we will provide a detailed introduction to this package through the following aspects.

1. Package use.
2. Package architecture description.
3. Package topic description.

Through the introduction of the three parts, it can help you:

1. Understand the package use.
2. Familiar with the file structure and function of the package.
3. Familiar with the topic related to the package for easy development and use.

Source code address: [https://github.com/RealManRobot/ros2\\_rm\\_robot.git](https://github.com/RealManRobot/ros2_rm_robot.git).

## 2. rm\_ros\_interface Package Use

This package does not have any executable commands, but it is used to provide the necessary message files for other packages.

## 3. rm\_ros\_interface Package Architecture Description

### 3.1 Overview of package files

The current rm\_driver package is composed of the following files.

— CMakeLists.txt	# compilation rule file
— include	# dependency header file folder
— rm_ros_interfaces	
— msg	# current message file (see below for details)
— Armoriginalstate.msg	
— Armsoftversion.msg	
— Armstate.msg	
— Cartepos.msg	
— Forcepositionmovejoint75.msg	
— Forcepositionmovejoint.msg	
— Forcepositionmovepose.msg	
— Force_Position_State.msg	
— Getallframe.msg	
— GetArmState_Command.msg	
— Gripperpick.msg	
— Gripperset.msg	
— Handangle.msg	
— Hand force.msg	
— Handposture.msg	
— Handseq.msg	
— Handspeed.msg	
— Jointerrclear.msg	



```
| |— Jointerrorcode75.msg
| |— Jointerrorcode.msg
| |— Jointpos75.msg
| |— Jointpos.msg
| |— Lift height.msg
| |— Liftspeed.msg
| |— Liftstate.msg
| |— Movec.msg
| |— Movej75.msg
| |— Movej.msg
| |— Movejp.msg
| |— Movel.msg
| |— Setforceposition.msg
| |— Setrealtimepush.msg
| |— Sixforce.msg
| |— Stop.msg
|— package.xml
|— src
```

# dependency declaration file

## 4. rm\_ros\_interface message description

### 1. Joint error code: Jointerrorcode.msg

```
uint16[] joint_error
uint8 dof
```

#### msg member

#### uint16[] joint\_error

Error message for each joint.

#### uint8 dof

Degree of freedom message of the robotic arm.

### 2. Clearing the joint's error code: Jointerrclear.msg

```
uint8 joint_num
bool block
```

#### msg member

#### joint\_num

the corresponding joint number, from the base to the robotic arm gripper, the number is 1-6 or 1-7.

#### block

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.



### 3. All coordinate system names: Getallframe.msg

```
string[10] frame_name
```

**msg member**

**frame\_name**

The array of work coordinate system names returned

### 4. Joined motion: Movej.msg

```
float32[] joint
```

```
uint8 speed
```

```
bool block
```

```
uint8 dof
```

**msg member**

**joint**

Joint angle, float type, unit: radians.

**speed**

Speed percentage ratio coefficient, 0-100.

**block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

**dof**

Degree of freedom message of the robotic arm.

### 5. Linear motion: MoveL.msg

```
geometry_msgs/Pose pose
```

```
uint8 speed
```

```
bool block
```

**msg member**

**pose**

Robotic arm pose: geometry\_msgs/Pose type, x, y, z coordinates (float type, unit: m) + quaternion (float type).

**speed**

Speed percentage ratio coefficient, 0-100.

**block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

### 6. Circular motion: Movec.msg



```
geometry_msgs/Pose pose_mid  
geometry_msgs/Pose pose_end  
uint8 speed  
bool block
```

#### **msg member**

##### **pose\_mid**

Middle pose: geometry\_msgs/Pose type, x, y, z coordinates (float type, unit: m) + quaternion.

##### **pose\_end**

Target pose: geometry\_msgs/Pose type, x, y, z coordinates (float type, unit: m) + quaternion.

##### **speed**

Speed percentage ratio coefficient, 0-100.

##### **block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

### **7. Joint space planning to target pose: Movejp.msg**

```
geometry_msgs/Pose pose  
uint8 speed  
bool block
```

#### **msg member**

##### **pose**

Target pose: geometry\_msgs/Pose type, x, y, z coordinates (float type, unit: m) + quaternion.

##### **speed**

Speed percentage ratio coefficient, 0-100.

##### **block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

### **8. Joint transmission: Jointpos.msg**

```
float32[] joint  
bool follow  
float32 expand  
uint8 dof
```

#### **msg member**





### **joint**

Joint angle, float type, unit: radians.

### **follow**

Follow state, bool type, true: high follow, false: low follow, default high follow if not set.

### **expand**

Expand joint, float type, unit: radians.

### **dof**

Degree of freedom message of the robotic arm.

## **9. Pose transmission: Cartepos.msg**

```
geometry_msgs/Pose pose
bool follow
```

### **msg member**

#### **pose**

Robotic arm poses geometry\_msgs/Pose type, x, y, z coordinates (float type, unit: m) + quaternion.

#### **follow**

Follow state, bool type, true: high follow, false: low follow, default high follow if not set.

## **10. Current robotic arm state (Angle + Euler angle): Armoriginalstate.msg**

```
float32[] joint
float32[6] pose
uint16 arm_err
uint16 sys_err
uint8 dof
```

### **msg member**

#### **joint**

Joint angle, float type, unit: °.

#### **pose**

Current pose of the robotic arm, float type, x, y, z coordinates, unit: m, x, y, z Euler angle, unit: degree.

#### **arm\_err**



Robotic arm running error code, unsigned int type.

#### **arm\_err**

Controller error code, unsigned int type.

#### **dof**

Degree of freedom message of the robotic arm.

### **11. Current arm state (radians + quaternion): Armstate.msg**

```
float32[] joint
geometry_msgs/Pose pose
uint16 arm_err
uint16 sys_err
uint8 dof
```

#### **msg member**

##### **joint**

Joint angle, float type, unit: radians.

##### **pose**

Current pose of the robotic arm, float type, x, y, z coordinates, unit: m, x, y, z, w quaternion.

##### **arm\_err**

Robotic arm running error code, unsigned int type.

##### **arm\_err**

Controller error code, unsigned int type.

##### **dof**

Degree of freedom message of the robotic arm.

### **12. Getting the software version: Armsoftversion.msg**

```
string plan version
string ctrlversion
string kernal1
string kernal2
string productversion
```

#### **msg member**

##### **planversion**

The read user interface kernel version number, string type.

##### **ctrlversion**



Real-time kernel version number, string type.

#### **kernal1**

The version number of sub-core 1 of the real-time kernel, string type.

#### **kernal2**

The version number of sub-core 2 of the real-time kernel, string type.

#### **productversion**

Robotic arm model, string type.

### **13. Gripper's pick: Gripperpick.msg**

```
uint16 speed
uint16 force
bool block
```

#### **msg member**

##### **speed**

Gripper pick speed, unsigned int type, range: 1-1000.

##### **force**

Gripper pick torque threshold, unsigned int type, range: 50-1000.

##### **block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

### **14. Gripper's pick (gripper's pick-on): Gripperpick.msg**

```
uint16 speed
uint16 force
bool block
```

#### **msg member**

##### **speed**

Gripper pick speed, unsigned int type, range: 1-1000.

##### **force**

Gripper picks torque threshold, unsigned int type, range: 50-1000.

##### **block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

### **15. Gripper reaching the given position: Gripperset.msg**

```
uint16 position
bool block
```



### **msg member**

#### **position:**

Gripper target position, unsigned int type, range: 1-1000, representing the degree of opening of the gripper: 0-70 mm.

#### **block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

### **16. Force-position mixing control: Setforceposition.msg**

```
uint8 sensor
uint8 mode
uint8 direction
int16 n
bool block
```

### **msg member**

#### **sensor**

Sensor; 0 - One-axis force; 1 - Six-axis force.

#### **mode**

Mode: 0 - Base coordinate system force control; 1 - Tool coordinate system force control.

#### **Direction**

Force control direction; 0 - Along the X-axis; 1 - Along the Y-axis; 2 - Along the Z-axis; 3 - Along the RX posture direction; 4 - Along the RY posture direction; 5 - Along the RZ posture direction.

#### **n**

Force value, unit: 0.1 N.

#### **block**

whether it is a blocking mode, true: blocking, false: non-blocking.

### **17. Six-axis force data: Sixforce.msg**

```
float32 force_fx
float32 force_fy
float32 force_fz
float32 force_mx
float32 force_my
float32 force_mz
```



**msg member**

**force\_fx**

the force along the x-axis direction.

**force\_fy**

the force along the y-axis direction.

**force\_fz**

the force along the z-axis direction.

**force\_mx**

the force when rotating along the x-axis direction.

**force\_my**

the force when rotating along the y-axis direction.

**force\_mz**

the force when rotating along the z-axis direction.

## **18. Setting the dexterous hand posture: Handposture.msg**

```
uint16 posture_num  
bool block
```

**msg member**

**posture\_num**

The serial number of the posture pre-saved in the dexterous hand, ranges from 1 to 40.

**block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

## **19. Setting the dexterous hand action sequence: Handseq.msg**

```
uint16 seq_num  
bool block
```

**msg member**

**seq\_num**

The serial number of the sequence pre-saved in the dexterous hand, ranging from 1 to 40.

**block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.



## 20. Setting the angles of various degrees of freedom for the dexterous hand:

### Handangle.msg

```
int16[6] hand_angle  
bool block
```

#### msg member

#### hand\_angle

Hand angle array, range: 0-1000. And -1 represents that no operation is performed on this degree of freedom and the current state remains.

#### block

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

## 21. Setting the dexterous hand action sequence: Handspeed.msg

```
uint16 hand_speed  
bool block
```

#### msg member

#### hand\_speed

Hand speed, range: 1-1000.

#### block

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

## 22. Setting the force threshold for the dexterous hand: Handforce.msg

```
uint16 hand_force  
bool block
```

#### msg member

#### hand\_force

Hand force, range: 1-1000.

#### block

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

## 23. Transmissive force-position mixing control compensation (angle):

### Forcepositionmovejoint.msg

```
float32[] joint  
uint8 sensor  
uint8 mode  
int16 dir  
float32 force
```



```
bool follow
uint8 dof
```

**msg member**

**joint**

Angle force-position mixing transmission, unit: radians.

**sensor**

Type of sensor used, 0 - One-axis force, 1 - Six-axis force.

**mode**

Mode, 0 - Along the work coordinate system, 1 - Along the tool end coordinate system.

**dir**

Force control direction, 0 to 5 represent X/Y/Z/Rx/Ry/Rz respectively, and the default direction for one-axis force type is the Z direction.

**force**

Force value, accuracy: 0.1 N or 0.1 Nm.

**block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

**dof**

Degree of freedom message of the robotic arm.

## 24. Transmissive force-position mixing control compensation (pose):

**Forcepositionmovejoint.msg**

```
geometry_msgs/Pose pose
uint8 sensor
uint8 mode
int16 dir
float32 force
bool follow
```

**msg member**

**pose**

Robotic arm pose message, x, y, z position message + quaternion posture message.

**sensor**



Type of sensor used, 0 - One-axis force, 1 - Six-axis force.

**mode**

Mode, 0 - Along the work coordinate system, 1 - Along the tool end coordinate system.

**dir**

Force control direction, 0 to 5 represent X/Y/Z/Rx/Ry/Rz respectively, and the default direction for one-axis force type is the Z direction.

**force**

Force value, accuracy: 0.1 N or 0.1 Nm.

**block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

**25. Speed open loop control (lifting mechanism): Liftspeed.msg**

```
int16 speed
bool block
```

**msg member**

**speed**

Speed percentage, -100-100. Speed < 0: the lifting mechanism moves downward; Speed > 0: the lifting mechanism moves upward; Speed = 0: the lifting mechanism stops.

**block**

whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

**26. Position closed-loop control (lifting mechanism): Lift height.msg**

```
uint16 height
uint16 speed
bool block
```

**msg member**

**height**

Target height, unit: mm, accuracy: 0-2600.

**speed**

Speed percentage, 1-100.

**block**





whether it is a blocking mode, bool type, true: blocking, false: non-blocking.

## 27. Getting the state of the lifting mechanism: Liftstate.msg

```
int16 height
int16 current
uint16 err_flag
```

**msg member**

**height**

Current lifting mechanism height, unit: mm, accuracy: 1mm, range: 0-2300.

**current**

Lifting drive error code, error code type refers to joint error code.

## 28. Getting (setting) UDP active reporting configuration: Setrealtimepush.msg

```
uint16 cycle
uint16 port
uint16 force_coordinate
string ip
```

**msg member**

**cycle**

Set the broadcast cycle, which is a multiple of 5ms.

**port**

Set the port number for broadcasting.

**force\_coordinate**

Coordinate system for external force data of the system, where 0 is the sensor coordinate system, 1 is the current work coordinate system, and 2 is the current tool coordinate system.

**IP**

Customized reporting target IP address.

## 29. Getting the state of the lifting mechanism: Liftstate.msg

```
int16 height
int16 current
uint16 err_flag
```

**msg member**

**height**

Current lifting mechanism height, unit: mm, accuracy: 1mm, range: 0-2300.



#### **current**

Lifting drive error code, error code type refers to joint error code.

### **30. Getting the state of the lifting mechanism: Liftstate.msg**

```
uint16 cycle
uint16 port
uint16 force_coordinate
string ip
```

#### **msg member**

##### **cycle**

Set the broadcast cycle, which is a multiple of 5ms (default 1 i.e.  $1 * 5 = 5$  ms, 200 Hz).

##### **port**

Set the port number for broadcasting (default 8089).

##### **force\_coordinate**

Set the coordinate system for external force data (only supported by robotic arms with force sensors).

##### **string ip**

Set the custom reporting target IP address (default 192.168.1.10).