User Manual

Segway RMP

(Robot Mobile Platform)

RMP Lite 220

General Information

The Robot Mobile Platform（RMP）provides general or integrated robot chassis solutions for enterprises or third-party developers, with versatility, durability and performance in mind. RMP Lite 220 (hereinafter referred to as RMP) is a mobile robot chassis product designed for indoor and outdoor distribution, inspection, service, cleaning, and warehousing AGV. It is designed to provide large-scale and customized services for companies in the robotics field such as special application robots.

Main features of RMP：

- Hardware: compact and equipped with a large-capacity battery;
- Software: compatible with ROS and Isaac operating systems;
- Hardware modular design, interface with software SDK, support secondary development or customized service;
- Support extension kits include: light strips, sensor mounting rods, and so on.

Safety

Incorrect use of RMP may cause loss of control, collision or fall of the RMP, resulting in property damage, personal injury, and even death. Therefore, in order to reduce risks and avoid injury, please read and follow all instructions and warnings in this manual.

The following secure messaging conventions are used in this document：

Warning！	Warns you of operations that may cause serious injury or even death
Attention！	Warns you of operations that may cause minor or moderate injuries
Please note	Indicates important information, but does not involve personal injury

Warning！

- Please keep RMP out of the reach of children and pets. Accidental movement of RMP may cause injury or even death.
- Please do not sit, stand or ride on the RMP. Doing so may cause injury or even death.
- Please do not control RMP to hit people or animals. Collision may cause injury or even death.

- When RMP is running, remind people nearby at all times. Accidental collision with RMP may cause injury or even death.
- Avoid power failure on slopes. RMP cannot maintain its position on the slope when the power is off. The power off will cause RMP to slide, which may cause injury or even death.
- RMP can accelerate quickly. It is recommended that using low speed to practice until the you are familiar with controlling RMP. Accidental movement of RMP may cause injury or even death.
- Please do not try to disassemble the battery, it may cause electric shock, burns or even fire. Attempting to open the battery case will damage the battery case, release toxic and harmful substances, and also render the battery unusable.
- The same with all rechargeable batteries, please do not charge near flammable materials, which may cause a fire.
- If the battery case is damaged or the battery emits peculiar smell, smoke, overheating or leakage, please do not continue to use the battery and do not contact with any substances leaking from the battery to avoid poisoning.
- Strictly observe and follow all safety information on the warning label on the battery. Failure to do so may result in injury or even death.

- Please do not use cables that have been seriously worn or damaged, which may shock yourself or damage the RMP.

Attention!

- The performance parameters should be set correctly and carefully. RMP follows the commands issued to it, users are responsible for implementing correct and safe performance parameters.
- Do not charge the battery may cause permanent damage to the battery.
- Only can use the charger that is provided with RMP Lite 220 to charge the batter.
- Before operating RMP, please be sure to read the user’s manual and be familiar with the operation of RMP and various precautions.

Please Note

- If the user modifies the chassis without communication with Segway-Ninebot and causes an accident, Segway-Ninebot does not assume any responsibility.

Table Of Contents

1 Product introduction

1.1Product diagram

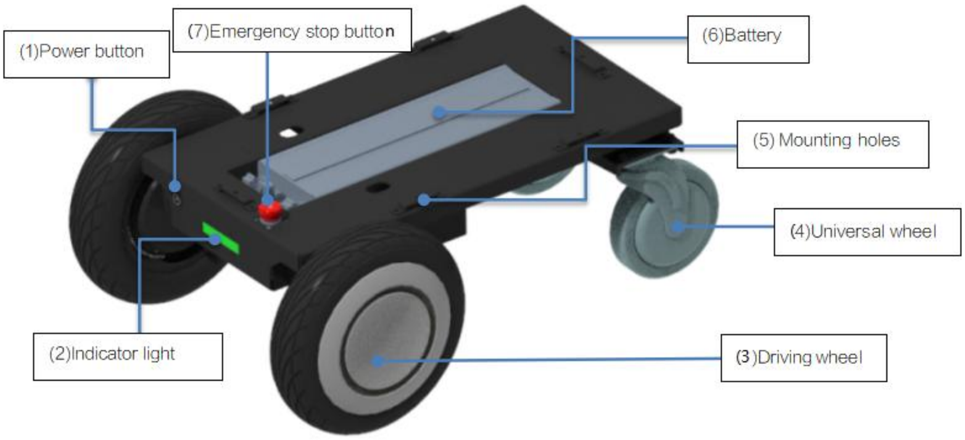


Figure 1

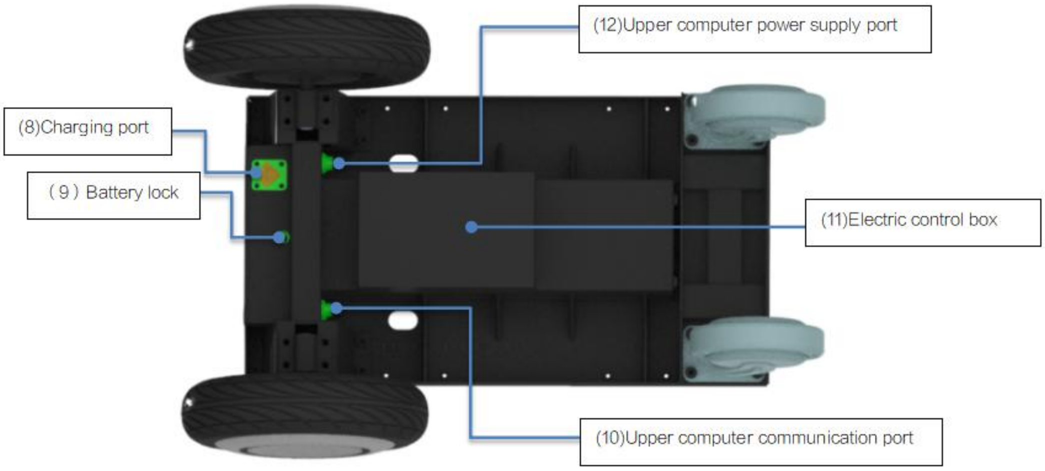

(12)Upper computer power supply port
(8)Charging port
(9) Battery lock
(11)Electric control box
(10)Upper computer communication port

Figure 2

1.2 Component description

Table 1

No.	Component name	Description
1	Power button	On：Hold down, when power button and indicator light are always on and accompanied by a beep, chassis boots successfully. At this time, the chassis is in lock mode, and the indicator light is steady yellow. Off：Hold down until the prompt sound starts，release the button，chassis shut down successfully. At this time, power button and indicator light are always off.
2	Indicator light	Colors and status of the indicator light represent different modes of the product.
3	Driving wheel	11 inch high adhesion pneumatic tire with good shock absorption and design with drainage tank.
4	Universal wheel	Super Artificial rubber, lightweight and shock absorption.
5	Mounting hole	Use to install upper equipment.
6	Battery	Power supply to chassis and upper system.
7	Emergency stop button	Use to switch the chassis to emergency stop mode in an emergency.
8	Charging port	Connect the charger to charge the device.
9	Battery lock	Used to fix the battery, need to use the matching key to open.
10	Upper computer power supply port	It supplies power to the upper computer, with a maximum current of 10A.
11	Electric control box	Use to install circuit module to control the chassis.
12	Upper computer power supply port	It supplies power to the upper computer.

1.3 Remote control

1.3.1 Remote control diagram

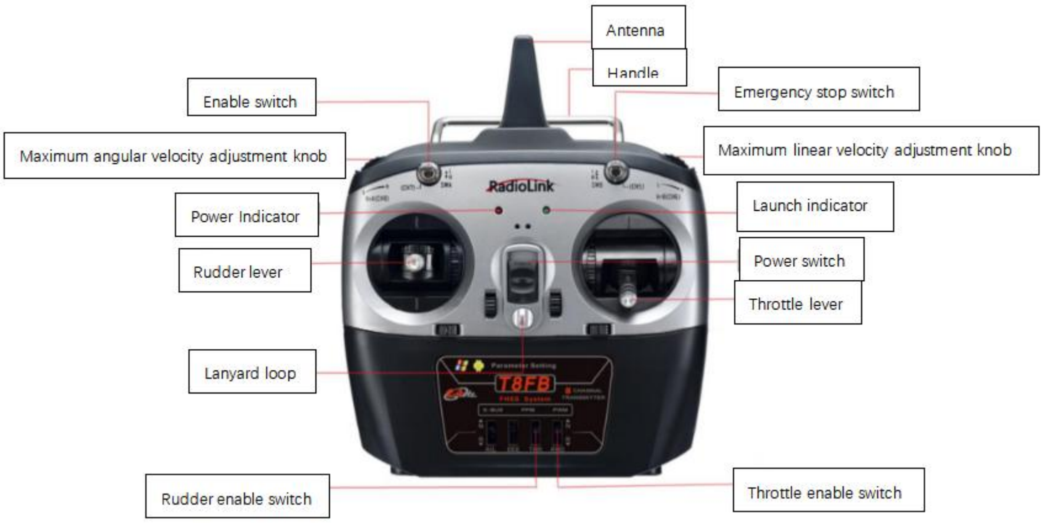


Figure 3

*The forward or backward of the remote control input (throttle or rudder) can be realized by flipping the enable switch under the T8FB.

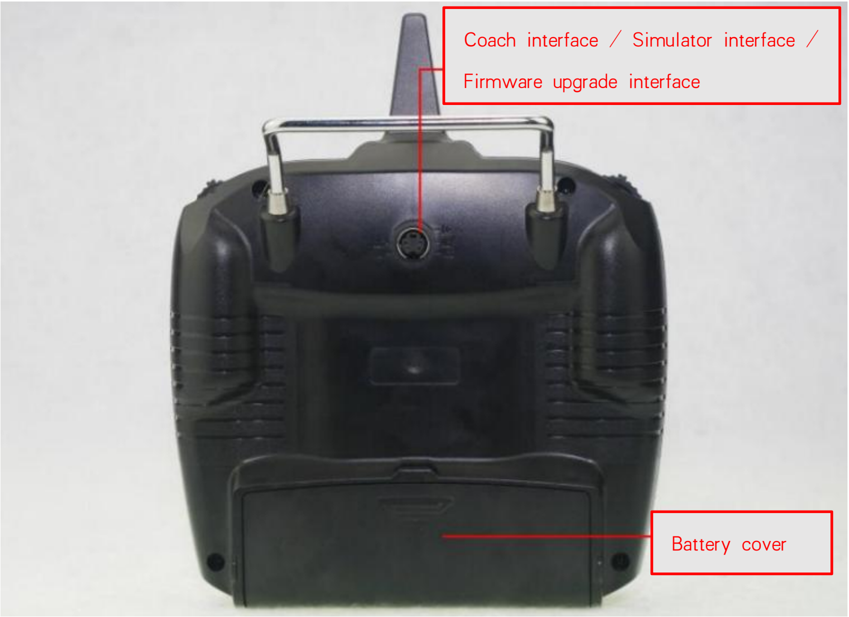

Coach interface / Simulator interface / Firmware upgrade interface
Battery cover

Figure 4

*The alarm voltage of the remote control is adaptive to 2S, 3S, 4S lithium batteries and 4 NI-MH batteries. That is, if T8FB is powered by 2S, 3S, 4S lithium batteries or 4 NI-HM batteries, after connecting the battery, T8FB will automatically set the low voltage alarm value according to the battery type.

1.3.2 Receiver pairing

Each transmitter has an independent ID code. Before starting to use the device, receiver must pair the code with the transmitter. After pairing the code, the ID code is stored in the receiver, and there is no need to pair it again, unless the receiver is used with another transmitter. When you have a new receiver, you must pair the code again, otherwise the receiver will not work normally.

（1）Place the remote control and receiver horizontally with a spacing of about 50cm;

（2）Turn on the power switch of the remote control to supply power to the receiver, and the receiver’s LED light starts to flash slowly;

（3）Press the pairing code button （ID SET）on the side of the receiver for more than 1 second, the LED light starts to flash quickly, that means the code is being paired, and the receiver will look for the nearest remote control to pair the code;

（4）When the receiver’s LED light stops flashing, it means that pairing the code is completed. If the receiver’s LED light flashes slowly, it means that pairing the code has failed, the code needs to be paired again.

1.3.3 Remote control control car instructions

（1）Turn on the RMP chassis: press the RMP power button;

Note: Please check the RMP status. Press and hold until the buzzer sounds and there is no continuous beeping, the indicator light is steady yellow.

（2）Turn on the remote control: push up the power switch of the remote control;

Note: ensure that the remote control is not in the emergency stop state and enters the enable state.That is, the emergency stop switch is not at the bottom, and the enable switch is dialed from the top to the bottom.

（3）At this time RMP is in Normal mode , see the table below for specific operations:

Table 2: Car control and remote control operation

Car control	Remote control operation
Turn left/ right	Rudder lever
Move forward/backward	Throttle lever
Emergency stop/ exit emergency stop	Emergency stop switch：At the top: exit the emergency stop; At the bottom: start the emergency stop
Adjust the maximum value of angular velocity	Maximum angular velocity adjustment knob: Turn left: the maximum angular velocity decreases, Turn right: increases.
Adjust the maximum value of linear velocity	Maximum linear velocity adjustment knob: Turn left: the maximum linear velocity decreases, Turn right: increases.
Enable/Disable	Turn the enable switch from the top to the bottom: Enable; Turn the enable switch from the bottom to the top: Disable.

1.3.4 Upper computer control car instructions

The host computer is the control computer, which can directly issue control commands and display various information changes on the screen. The upper computer controls the lower computer and provides some necessary operating environment for the lower computer, and extends the man-machine control or demonstration function that the lower computer can provide. The upper computer has the characteristics of leading management, coordinating resources, monitoring agency, and controlling RMP.

（1）Turn on the RMP chassis: press the RMP power button;

Note 1 : Please check the RMP status. Press and hold until the buzzer sounds and there is no continuous beeping, the indicator light is steady yellow.

Note 2: When the upper computer controls the car, the remote control cannot be turned on. Or if the remote control is turned on, turn its enable switch upward.

（2）Ensure that the RMP serial line or CAN line is connected to the upper computer；

（3）In the upper computer, give permissions to “/sdcard/ segway/ hardware_log/” folder, otherwise it will fail to create a new log file; give permissions to all files in the “/catkin_ws/ src/ RosCode/ segwayrmp/ lib/” directory（no need to reset after first setup）：

```
`cd /sdcard/segway/hardware_log`

`sudo chmod 777 /sdcard/segway/hardware_log/`

`cd $PRO_HOME$/catkin_ws/src/RosCode/segwayrmp/lib/`

`sudo chmod 777 *
```

（4）In ”catkin_ws/ src/ RosCode/ segwayrmp/ Cmakelists.txt” file，according to the upper computer in x86_ 64 or arm platform, select the compilation option,as shown below when compiling in x86_64 platform, comment out “libctrl_arm64-v8a.so” with the symbol “#”，（no need to reset after first setup）:

`target_link_libraries(SmartCar`

`${catkin_LIBRARIES}`

`#${PROJECT_SOURCE_DIR}/lib/libctrl_arm64-v8a.so //in x86_64 platform， comment out this line， in arm platform, do not comment out this line`

`${PROJECT_SOURCE_DIR}/lib/libctrl_x86_64.so //in arm platform，comment out this line，in x86_64 platform, do not comment out this line`

（5）Enter ROS system，run the following command to compile the “segway_msgs” package message.

```
cd catkin_ws
catkin_make
-DCATKIN_WHITELIST_PACKAGES='segway_msgs'
```

（6）Enter ROS system，run the following command to compile the “segwayrmp” package message.

```
cd catkin_ws
catkin_make
-DCATKIN_WHITELIST_PACKAGES='segwayrmp'
```

（7）Control car in ROS system:

1）Create a new terminal，run the following command:

```
cd catkin_ws
```

```
roscore
```

2）Create a new terminal，run the following command, run SmarCar node:

```
cd catkin_ws
source devel/setup.bash
rosrun segwayrmp SmartCar
```

3）Create a new terminal，run the following command, run routine test node:

```
cd catkin_ws
source devel/setup.bash
rosrun segwayrmp ChassisResponseTest
```

2 Software introduction

This chapter introduces relevant documents, software interface functions and fault code information provided by RMP.

2.1 Documents provided to users

Table 3 Documents provided

Document	Function
Libctrl_x86_64.so	Provide C/C++ chassis-related interfaces in x86 platform
Libctrl_arm64-v8a.so	Provide C/C++ chassis-related interfaces in arm platform
Comm_ctrl_navigation.h	C/C++ API interface header file
ROS package	Provide chassis control ROS nodes

2.2 Interface function introduction

2.2.1 C/C++ Interface introduction

Table 4 callback data type

Callback type	Callback No.	Function description	Data structure

Chassis_Data_Speed	1	Chassis speed information	typedef struct{
			int16_t l_speed;

			int16_t r_speed;
			int16_t car_speed;
			int16_t turn_speed;
			}chassis_speed_data_t;

Chassis_Data_Ticks	2	Chassis encoder information	typedef struct{
			int32_t l_ticks;
			int32_t r_ticks;
			}motor_ticks_t;

Chassis_Data_Odom_Pose_xy	3	Odom pose information	typedef struct{
			float pos_x;
			float pos_y;
			}odom_pos_xy_t;

Chassis_Data_Odom_Euler_xy	4	Odom Euler x/y axis information	typedef struct{
			float euler_x;
			float euler_y;
			}odom_euler_xy_t;

Chassis_Data_Odom_Euler_z	5	Odom Euler Z axis information	typedef struct{
			float euler_z;
			}odom_euler_z_t;

Chassis_Data_Odom_Linevel_xy	6	Odom speed x/y axis information	typedef struct{
			float vel_line_x;
			float vel_line_y;
			}odom_vel_line_xy_t;

Chassis_Data_Imu_Gyr	7	Gyroscope data	typedef struct{
			int16_t gyr[3];
			}imu_gyr_original_data_;

Chassis_Data_Imu_Acc	8	Accelerometer data	typedef struct{
			int16_t acc[3];
			}imu_acc_original_data_;

Note 1：Odom data：The default heading angle is 0 degrees at start up.

Note 2：IMU（gyroscope and accelerometer）data：the carrier coordinate system XYZ corresponds to the right front up.

Table 5 event definition

Event type	Event No.	Function description
ChassisBootReadyEvent	1	Chassis central control

		board start up completed
PadPowerOffEvent	2	Chassis shutdown
OnEmergeStopEvent	3	Enter emergency stop
OutEmergeStopEvent	4	Exit emergency stop
OnLockedRotorProtectEvent	5	Locked rotor event occurs
OutLockedRotorProtectEvent	6	Locked rotor event removes
OnLostCtrlProtectEvent	7	Lost control event occurs
OutLostCtrlProtectEvent	8	Lost control event removes
CalibrateGyroSuccess	9	Calibrate the gyroscope success
CalibrateGyroFail	10	Calibrate the gyroscope fail
CalibratePasheCurrentSuccess	11	Calibrate phase current success
CalibratePasheCurrentFail	12	Calibrate phase current fail

Table 6 get/set interface

Interface name	Interface description
get_err_state	Get the error code of the upper computer/central control board/motor board/battery
get_bat_soc	Get the percentage of battery remaining
get_bat_charging	Get battery charge status（1: charging; 0: non-charging）
get_bat_mvol	Get battery voltage (unit: millivolt (mV))
get_bat_mcurrent	Get battery current (unit: mA)
get_bat_temp	Get battery temperature (unit: degrees Celsius (° C))
get_chassis_work_model	Get chassis work model（0：Unload；1：Onload）
get_chassis_load_state	Get chassis load state （0：empty；1：full）
get_chassis_mode	Get chassis mode（0：Locked； 1：Control； 2：Push；3：Emergency stop; 4：Error）
get_ctrl_cmd_src	Get the current chassis control source (0: remote control; 1: upper computer)
get_vehicle_meter	Get chassis mileage (unit: meter(m))
get_host_version	Get the upper computer version number
get_chassis_central_version	Get the control board version number
get_chassis_motor_version	Get the motor board version number（Reserved）
get_line_forward_max_vel_fb	Get the forward speed limit value of the chassis (unit: meter per hour(m/h))
get_line_backward_max_vel_fb	Get the backward speed limit value of the chassis (unit: meter per hour(m/h))
get_angular_max_vel_fb	Get the limit value of chassis angular velocity (unit:

	milliradian per second(mrad/s))
getIapTotalProgress	Get IAP progress
iapCentralBoard	Upgrade the central control board IAP
iapMotorBoard	Upgrade the motor board IAP
isHostIapOver	Check if IAP is over
getHostIapResult	Get the IAP result (3: completed; 4: failed: 5: interrupted; 0: meaningless)
getHostIapErrorCode	Get IAP error code
get_chassis_hang_mode	Get whether the chassis is in the hang mode (0: not in the hang mode; 1: in the hang mode)
get_charge_mos_ctrl_status	Get charging MOS status (1: charging MOS is on, 0: MOS is off) (temporarily reserved)
set_cmd_vel	Set the chassis linear velocity and angular velocity (unit: meter per second(m/s) and radian per second(rad/s))
set_line_forward_max_vel	Set the forward speed limit value of the chassis (unit: meter per second(m/s))
set_line_backward_max_vel	Set the backward speed limit value of the chassis (unit: meter per second(m/s))
set_angular_max_vel	Set the limit value of chassis angular velocity (unit: radian per second(rad/s))
set_enable_ctrl	Set the enable state of the upper computer control car on the chassis (1: enable; 0 disable)
init_control_ctrl	Chassis initialization interface
exit_control_ctrl	Chassis exit initialization interface
set_smart_car_serial	Set the serial port name used by the upper computer dynamic library
set_comu_interface	Set the communication interface for communication with the chassis (0: serial port; 1: CAN)
set_chassis_load_state	Set chassis load state (0：empty；1：full)
set_chassis_poweroff	Set chassis shutdown command
set_remove_push_cmd	Remove chassis push command
setHostIapCanceled	Cancel the upper computer IAP command
set_chassis_hang_mode	Set chassis hang mode（1：enter the hang mode；0：exit the hang mode）
set_charge_mos_ctrl	Set charging MOS switch（1：turn on MOS, 0: turn off MOS）(temporarily reserved)

2.2.2 ROS Interface introduction—SmartCar

Table 7 news release

Topic Name	Function Description	Message Type	Message Type Info	Frequency

Bms_fb	Battery related information	Segway_msgs/ Bms_fb	int16 bat_soc int16 bat_charging int32 bat_vol int32 bat_current int16 bat_temp	1
Chassis_ctrl_src_fb	Chassis control command source	Segway_msgs/ Chassis_ctrl_src_fb	uint16 chassis_ctrl_cmd_src	1
Chassis_mileage_meter_fb	Chassis mileage	Segway_msgs/ Chassis_mileage_meter_fb	uint32 vehicle_meters	1
Chassis_mode_fb	Chassis mode	Segway_msgs/ Chassis_mode_fb	uint16 chassis_mode	1
Error_code_fb	Chassis error code	Segway_msgs/ Error_code_fb	uint32 host_error uint32 central_error uint16 left_motor_error uint16 right_motor_error uint32 bms_error	1
Motor_work_mode_fb	Chassis working mode	Segway_msgs/ Motor_work_mode_fb	uint16 motor_work_mode #0: no output torque 1: output torque	1
Speed_fb	Chassis speed	Segway_msgs/ Speed_fb	float32 car_speed float32 turn_speed float32 l_speed float32 r_speed uint64 speed_timestamp	40
Ticks_fb	Chassis encoder information	Segway_msgs/ Ticks_fb	int32 l_ticks int32 r_ticks uint64 ticks_timestamp	40
Odom	Odom data	Nav_msgs/odom		40
Imu	Imu data	Sensor_msgs/imu		40

Table 8 News Subscription

TopicName	Function Description	Message Type	Message Type Info
Cmd_vel	Control	Geometry_msgs/twist	Angular.z //rad/s

	chassis movement		Linear.x //m/s

Table 9 service client

Service name	Function Description	Message type	Message type info
chassis_send_event_srv	Send time number	Segway_msgs/chassis_send_event	chassis_send_event_id ros_is_received

Table 10 Service server

Service name	Function Description	Message type	Message type info
ros_get_load_param_cmd_srv	Get load parameter	Segway_msgs/ros_get_load_param_cmd	ros_get_load_param --- get_load_param #0:no_load, 1: full_load
ros_get_charge_mos_ctrl_status_cmd.srv	Get chassis charging MOS status (reserved temporarily)	Segway_msgs/ros_get_charge_mos_ctrl_status_cmd	ros_get_chassis_charge_ctrl_status # 1: MOS opened; 0: MOS closed --- chassis_charge_ctrl_status
ros_get_sw_version_cmd_srv	Get software version	Segway_msgs/ros_get_sw_version_cmd	ros_get_sw_version_cmd --- uint16 host_version uint16 central_version uint16 motor_version
ros_get_vel_max_feedback_cmd_srv	Get the maximum speed limit	Segway_msgs/ros_get_vel_max_feedback_cmd	ros_get_vel_max_fb_cmd --- forward_max_vel_fb backward_max_vel_fb angular_max_vel_fb
ros_set_charge_mos_ctrl_cmd.srv	Set chassis charging MOS (reserved temporarily)	Segway_msgs/ros_set_charge_mos_ctrl_cmd	ros_set_chassis_charge_ctrl --- chassis_set_charge_ctrl_result # 1: MOS opened; 0: MOS closed
ros_set_chassis_enable_cmd_srv	Set chassis enable command	Segway_msgs/ros_set_chassis_enable_cmd	ros_set_chassis_enable_cmd --- chassis_set_chassis_enable

			_result
ros_set_chassis_poweroff_cmd_srv	Set chassis shutdown command	Segway_msgs/ros_set_chassis_poweroff_cmd	ros_set_chassis_poweroff_cmd --- chassis_set_poweroff_result
ros_set_load_param_cmd_srv	Set chassis load state	Segway_msgs/ros_set_load_param_cmd	ros_set_load_param #0:no_load, 1: full_load --- chassis_set_load_param_result
ros_set_remove_push_cmd_srv	Set remove chassis push command	Segway_msgs/ros_set_remove_push_cmd	ros_set_remove_push_cmd --- chassis_set_revove_push_result
ros_set_vel_max_cmd_srv	Set the maximum speed limit	Segway_msgs/ros_set_vel_max_cmd_srv	ros_set_forward_max_vel ros_set_backward_max_vel ros_set_angular_max_vel --- chassis_set_max_vel_result

Table 11 Action server

Action name	Function Description	Message type	Message type info
ros_set_iap_cmd_action	Upgrade the board firmware IAP	Segway_msgs/ros_set_iap_cmdAction	Bool central_board_iap_enable --- Int16 iap_result #3: iap_state_complete; 4: iap_state_fail; 5: iap_state_abort Int16 error_code #When iap_result value is 4, this value represents the error code --- Int16 iap_percent

2.2.3 Error code information table

The error code is obtained through:

“uint32_t get_err_state(board_name_e board_name)” interface, and the corresponding information is as follows:

Table 12 Error code

Board name	Bit	Error info
host	0x00000000	No error
	0x00000001	Loss of control board
	0x00000002	Unplug the serial port module
Central	0x00000000	No error
	0x00000001	Car control command communication interrupted
	0x00000002	Motor board communication interrupted
	0x00000004	IMU initialization failed
	0x00000008	IMU failed to read data
	0x00000010	Lost control
	0x00000020	Locked rotor
	0x00000040	Failed to calibrate IMU
	0x00000080	Read Flash failed
	0x00000100	IMU data update failed
	0x00000200	Bms initialization failed to enter test mode
	0x00000400	Rollover
	0x00000800	Any motor board restart is detected
	0x00001000	Left magnetic encoder fault
	0x00002000	Right magnetic encoder fault
	0x00004000	Battery communication interrupted
	0x00008000	Battery communication interrupted（30s）
Motor	0x00000000	No error

	0x00000001	Phase current fault
	0x00000002	Phase voltage fault
	0x00000004	Lack of phase
	0x00000008	Under voltage
	0x00000010	Over voltage
	0x00000020	Over current
	0x00000040	Over temperature
	0x00000080	Locked rotor
	0x00000100	Electrical angle fault
	0x00000200	Excessive power fault
	0x00000400	Over speed fault
	0x00000800	Rotational speed sensor fault
	0x00001000	Angle sensor fault
	0x00002000	Current loop fault
	0x00004000	Speed loop fault
	0x00008000	Angle loop fault
Battery	0x00000000	No error
	0x00000008	Overcharge
	0x00000010	Charge over temperature protection

3 Firmware upgrade and version upgrade

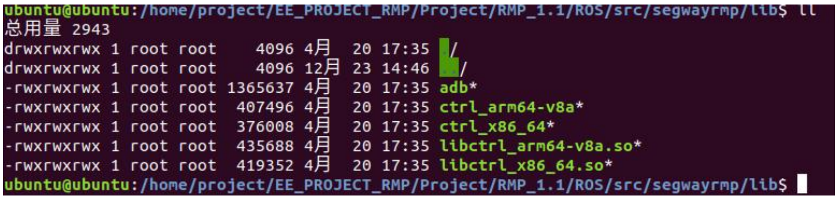
IAP is a software function module of the system, that is, in application programming, that means upgrade the single chip computer program online. This function uses the upper computer to burn the new version bin file to the single chip computer (including the central control board, the motor drive board, etc.) when the program is running. The premise is that the single chip computer’s bin file, which will be burned, needs to be named according to the requirements of the upper computer,and places it under the “/sdcard/firmware/” path of the upper computer. And then the bin file can be upgraded online through the command at the terminal.

3.1 Firmware upgrade

Before the firmware upgrade, it is necessary to test the data communication between the upper computer and each lower computer to check whether the communication is normal. Use the command to test in the shell terminal.

（1）View the path of the upper computer program

Enter the path where the upper computer program is located, and check whether the upper computer executable file exists. As shown in the figure below, they are arm executable file, x86 executable file, arm dynamic library, and x86 dynamic library:



```
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Project/RMP_1.1/ROS/src/segwayrmp/lib$ ll
总用量 2943
drwxrwxrwx 1 root root    4096 4月  20 17:35 ./
drwxrwxrwx 1 root root    4096 12月 23 14:46 ../
-rwxrwxrwx 1 root root 1365637 4月  20 17:35 adb*
-rwxrwxrwx 1 root root  407496 4月  20 17:35 ctrl_arm64-v8a*
-rwxrwxrwx 1 root root  376008 4月  20 17:35 ctrl_x86_64*
-rwxrwxrwx 1 root root  435688 4月  20 17:35 libctrl_arm64-v8a.so*
-rwxrwxrwx 1 root root  419352 4月  20 17:35 libctrl_x86_64.so*
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Project/RMP_1.1/ROS/src/segwayrmp/lib$
```

Figure 5

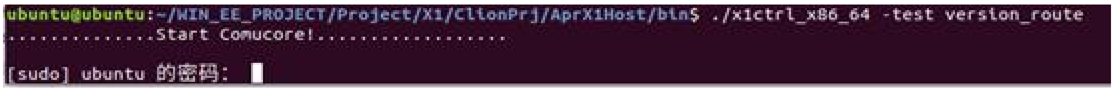
（2）Check the software version of each lower computer board

Check the software version of the lower computer. This step can test the communication between the upper computer and the lower computer at the same time. If the software version of each section of the lower computer can be checked through the upper computer, it indicates that the communication is normal.

Central control board test command：

./ctrl_x86_64s －test central

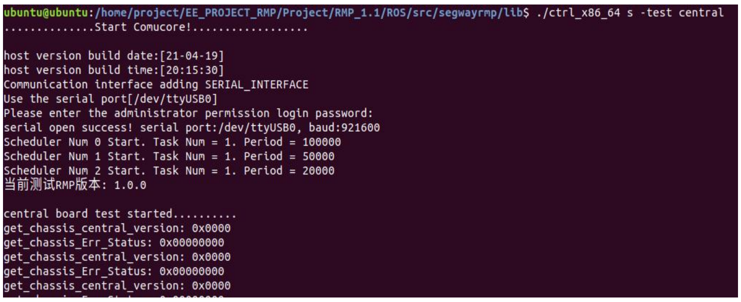
1）When connecting for the first time, if the serial port’s USB port does not have execution permission, the program requires root permission to modify the executable permission of the serial port’s USB port. At this time, you need to enter the system login password, and then hit the enter key, as shown in the figure below:



```
ubuntu@ubuntu:~/WIN_EE_PROJECT/Project/X1/ClionPrj/AprX1Host/bin$ ./x1ctrl_x86_64 -test version_route
..............Start Comucore!..................

[sudo] ubuntu 的密码: 
```

Figure 6

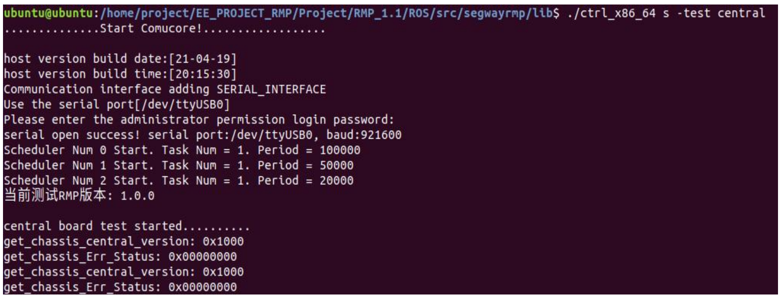
2）When communication fails, the version number is 0，as shown in the figure below:



```
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Project/RMP_1.1/ROS/src/segwayrmp/lib$ ./ctrl_x86_64 s -test central
..............Start Comucore!.................

host version build date:[21-04-19]
host version build time:[20:15:30]
Communication interface adding SERIAL_INTERFACE
Use the serial port[/dev/ttyUSB0]
Please enter the administrator permission login password:
serial open success! serial port:/dev/ttyUSB0, baud:921600
Scheduler Num 0 Start. Task Num = 1. Period = 100000
Scheduler Num 1 Start. Task Num = 1. Period = 50000
Scheduler Num 2 Start. Task Num = 1. Period = 20000
当前测试RMP版本: 1.0.0

central board test started.........
get_chassis_central_version: 0x0000
get_chassis_Err_Status: 0x00000000
get_chassis_central_version: 0x0000
get_chassis_Err_Status: 0x00000000
get_chassis_central_version: 0x0000
```

Figure 7

3）When the communication is successful, the version number is printed as follows, and it is a non-zero number. At this time, the communication between the upper computer and the single chip computer is normal, and the online upgrade can be performed:



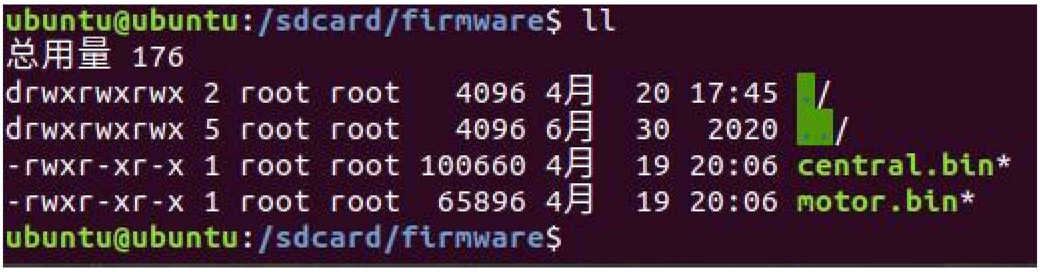
```
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Project/RMP_1.1/ROS/src/segwayrmp/lib$ ./ctrl_x86_64 s -test central
..............Start Comucore!.................

host version build date:[21-04-19]
host version build time:[20:15:30]
Communication interface adding SERIAL_INTERFACE
Use the serial port[/dev/ttyUSB0]
Please enter the administrator permission login password:
serial open success! serial port:/dev/ttyUSB0, baud:921600
Scheduler Num 0 Start. Task Num = 1. Period = 100000
Scheduler Num 1 Start. Task Num = 1. Period = 50000
Scheduler Num 2 Start. Task Num = 1. Period = 20000
当前测试RMP版本: 1.0.0

central board test started.........
get_chassis_central_version: 0x1000
get_chassis_Err_Status: 0x00000000
get_chassis_central_version: 0x1000
get_chassis_Err_Status: 0x00000000
```

Figure 8

3.2 Version upgrade

（1）Single chip computer bin file placement

Put the board’s bin file which will be upgraded into the “/sdcard/firmware” path of the upper computer, for example, the central control board’s bin file “central.bin”, as shown in the following figure:

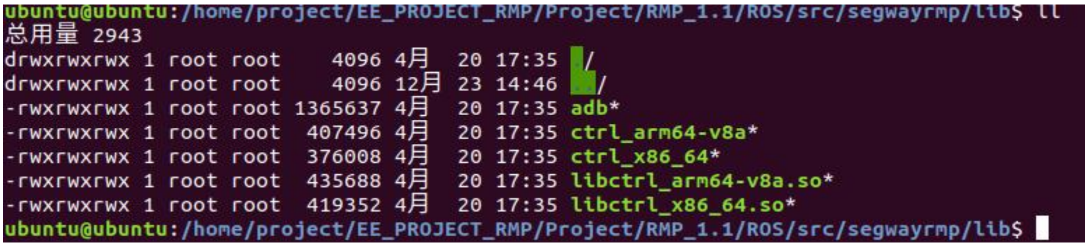


```
ubuntu@ubuntu:/sdcard/firmware$ ll
总用量 176
drwxrwxrwx 2 root root   4096 4月  20 17:45 ./
drwxrwxrwx 5 root root   4096 6月  30  2020 ../
-rwxr-xr-x 1 root root 100660 4月  19 20:06 central.bin*
-rwxr-xr-x 1 root root  65896 4月  19 20:06 motor.bin*
ubuntu@ubuntu:/sdcard/firmware$
```

Figure 9

（2）Online burning of bin file in lower computer

Enter the path where the upper computer program executable file “ctrl_x86_64” or “ctrl_arm64-v8a” is located, as follows:



```
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Project/RMP_1.1/ROS/src/segwayrmp/lib$ ll
总用量 2943
drwxrwxrwx 1 root root    4096 4月  20 17:35 ./
drwxrwxrwx 1 root root    4096 12月 23 14:46 ../
-rwxrwxrwx 1 root root 1365637 4月  20 17:35 adb*
-rwxrwxrwx 1 root root  407496 4月  20 17:35 ctrl_arm64-v8a*
-rwxrwxrwx 1 root root  376008 4月  20 17:35 ctrl_x86_64*
-rwxrwxrwx 1 root root  435688 4月  20 17:35 libctrl_arm64-v8a.so*
-rwxrwxrwx 1 root root  419352 4月  20 17:35 libctrl_x86_64.so*
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Project/RMP_1.1/ROS/src/segwayrmp/lib$
```

Figure 10

The commands for online upgrade of each board are as follows. After entering the upper computer program path, execute the following commands (use ‘s’ when using the serial port; use ‘c’ when using the CAN port):

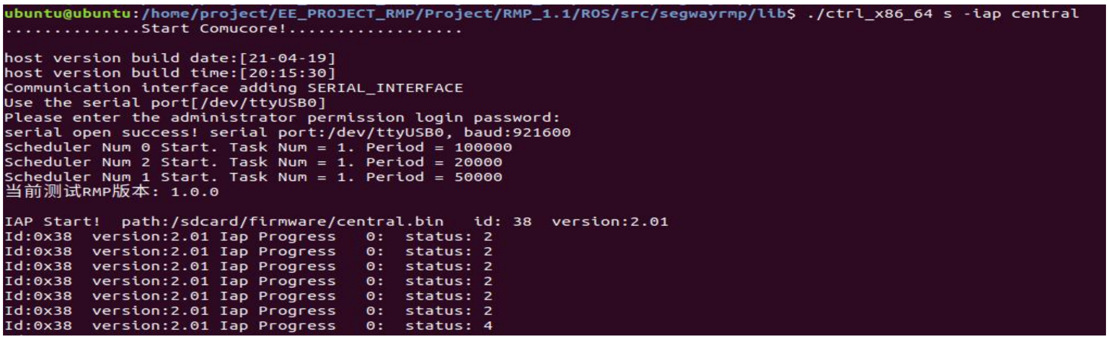
Central control board upgrade command：

“./ctrl_x86_64 s –iap central”

Motor board upgrade command：

“./ctrl_x86_64 s –iap motor”

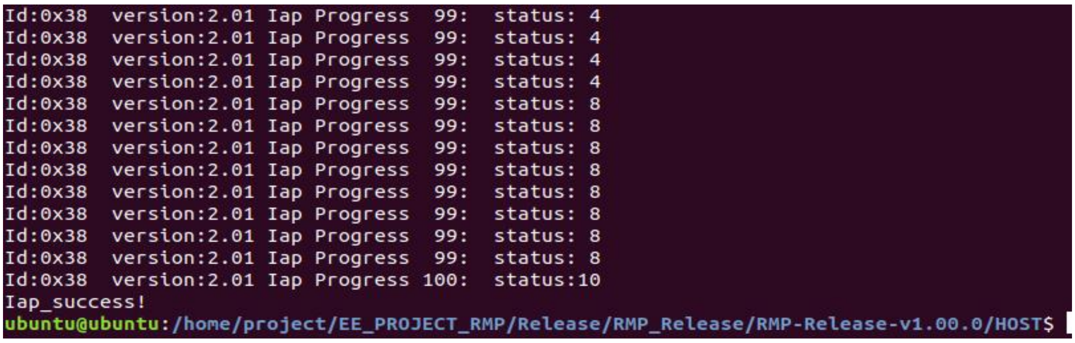
Take the central control board as an example, enter the command: “./ ctrl_x86_64 s –iap central” to upgrade, as shown in the figure below：



```
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Project/RMP_1.1/ROS/src/segwayrmp/lib$ ./ctrl_x86_64 s -iap central
..............Start Comucore!..................

host version build date:[21-04-19]
host version build time:[20:15:30]
Communication interface adding SERIAL_INTERFACE
Use the serial port[/dev/ttyUSB0]
Please enter the administrator permission login password:
serial open success! serial port:/dev/ttyUSB0, baud:921600
Scheduler Num 0 Start. Task Num = 1. Period = 100000
Scheduler Num 2 Start. Task Num = 1. Period = 20000
Scheduler Num 1 Start. Task Num = 1. Period = 50000
当前测试RMP版本: 1.0.0

IAP Start!  path:/sdcard/firmware/central.bin   id: 38  version:2.01
Id:0x38  version:2.01 Iap Progress   0:  status: 2
Id:0x38  version:2.01 Iap Progress   0:  status: 2
Id:0x38  version:2.01 Iap Progress   0:  status: 2
Id:0x38  version:2.01 Iap Progress   0:  status: 2
Id:0x38  version:2.01 Iap Progress   0:  status: 2
Id:0x38  version:2.01 Iap Progress   0:  status: 2
Id:0x38  version:2.01 Iap Progress   0:  status: 4
```

Figure 11

During the upgrading, you can view the upgrade progress. Progress represents the percentage of the IAP upgrade progress. When the Progress value reaches 100, it means that the routing board bin file has been programmed into the central control board chip. As shown below：



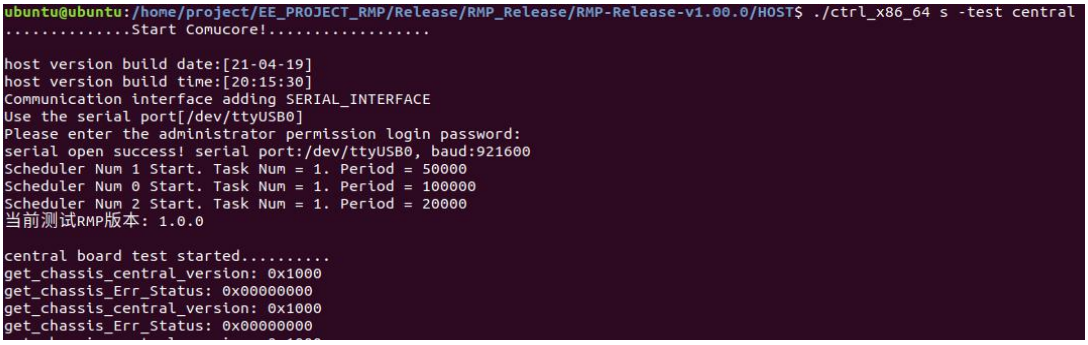
```
Id:0x38  version:2.01 Iap Progress  99:  status: 4
Id:0x38  version:2.01 Iap Progress  99:  status: 4
Id:0x38  version:2.01 Iap Progress  99:  status: 4
Id:0x38  version:2.01 Iap Progress  99:  status: 4
Id:0x38  version:2.01 Iap Progress  99:  status: 8
Id:0x38  version:2.01 Iap Progress  99:  status: 8
Id:0x38  version:2.01 Iap Progress  99:  status: 8
Id:0x38  version:2.01 Iap Progress  99:  status: 8
Id:0x38  version:2.01 Iap Progress  99:  status: 8
Id:0x38  version:2.01 Iap Progress  99:  status: 8
Id:0x38  version:2.01 Iap Progress  99:  status: 8
Id:0x38  version:2.01 Iap Progress  99:  status: 8
Id:0x38  version:2.01 Iap Progress 100:  status:10
Iap_success!
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Release/RMP_Release/RMP-Release-v1.00.0/HOST$
```

Figure 12

（3）Test the online upgrade result of the IAP version：

Perform step 1, test and check the software version number, enter the command: “./ ctrl_x86_64 s –test central”, in the path where the upper computer program is located, as shown below：

At this time, the software version number of the central control board is 0x1000, indicating that the online upgrade has been successful, and the communication between the upper computer and the central control board is good.



```
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Release/RMP_Release/RMP-Release-v1.00.0/HOST$ ./ctrl_x86_64 s -test central
..............Start Comucore!.................

host version build date:[21-04-19]
host version build time:[20:15:30]
Communication interface adding SERIAL_INTERFACE
Use the serial port[/dev/ttyUSB0]
Please enter the administrator permission login password:
serial open success! serial port:/dev/ttyUSB0, baud:921600
Scheduler Num 1 Start. Task Num = 1. Period = 50000
Scheduler Num 0 Start. Task Num = 1. Period = 100000
Scheduler Num 2 Start. Task Num = 1. Period = 20000
当前测试RMP版本: 1.0.0

central board test started.........
get_chassis_central_version: 0x1000
get_chassis_Err_Status: 0x00000000
get_chassis_central_version: 0x1000
get_chassis_Err_Status: 0x00000000
```

Figure 13

Appendix I: System parameters and mode switching logic

Table 1 System parameters

Structural parameter	Size	Length*Width*Height（mm） 730*500*280
	Structural parameters	Wheelbase*Tread*Ground clearance（mm）： 513.5*414*68
	Tire size	11Inch（280mm）hub motor
	Weight (with battery)	27.2kg
	Nominal load	50kg
	Obstacle surmounting	4cm/8° Slope/Deceleration zone
	Overhang	4mm Rear overhang
	Gear train structure	Front drive, differential steering
	Protection level	IPX5
Performance parameter	Maximum speed	3m/s
	Maximum steering speed	3rad/s
	Minimum turning radius	0m
	Braking distance	No load: 3m/s 0.95m，Braking acceleration: 0
	Control mode	Remote control, Upper computer control
	Braking mode	Electric brake
Communication	Communication Interface	UART，CAN
	Support driver、API	C/C++、ROS
	Feedback data	Magnetic， Hall，IMU
Battery	Battery	48V 24Ah
	Charging method	Manual line charging/Quick battery change
Interactive	Button	Emergency stop button, Power button
	Status indication	Power status indicator, Chassis status indicator, Control source indication, Battery display, Charging status display

Table 2 Mode switching logic

Chassis mode	Enter	Implement	Exit
Locked car mode	1. The default mode of chassis power on 2. Default mode after emergency stop recovery 3. In control car mode, when recoverability abnormality (such as communication timeout, communication disconnection, etc.) occurs, enters the locked car mode.	0 speed closed loop, shield speed command, and the status indicator light keeps yellow	1. An unrecoverable exception errorcode is detected and enters the error mode 2. Press the emergency stop button to enter the emergency stop mode
Control car mode	1. In locked car mode, the enable command is received 2. When the load exceeds 20kg, it is recommended to use ‘set_chassis_load_state’ switch the car control mode to overload mode	Closed loop, accepting control commands. Remote control car: indicator light flashes green; upper computer controls car: indicator light keeps green	1. An unrecoverable exception errorcode is detected and enters the error mode 2. when recoverability abnormality (such as communication timeout, communication disconnection, etc.) occurs, enters to locked car mode. 3. Press the emergency stop button to enter the emergency stop mode
Emergency stop mode	In non abnormal mode, press the emergency stop button	Relief force, shield speed and enable	The emergency stop button pops up and enters the locked car

		command, the status indicator light flashes red	mode
Error mode	Unrecoverable exception errorcode detected	Braking, releasing force, shielding speed and enable commands. The indicator light is keeps red	1. Restart

Appendix II Connector welding instructions

(I) Preparatory work

1、 Tools

Electric soldering iron, solder wire

2、 Materials

8pin connector, 2pin connector, 2 AWG16 cables, 8 AWG26 cables, as shown in Figure 1.

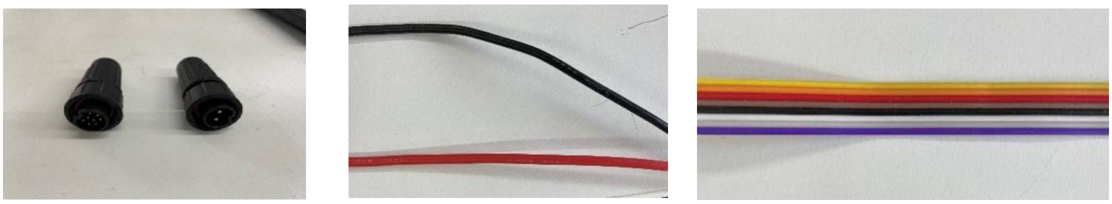


Figure 1

(II) Welding instructions (take 8pin connector as an example)

1. Figure 2 shows the 8pin connector received by the customer. Screw the connector from the position shown by the red arrow to disassemble it into the state shown in Figure 3;

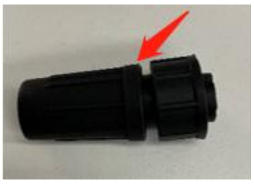


Figure 2

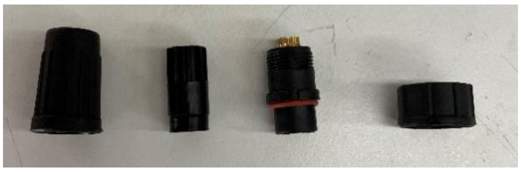


Figure 3

2. Take out the part shown in Figure 4, which is the part that needs to be welded;

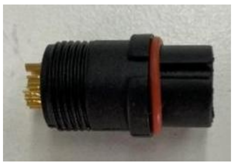


Figure 4

3. As shown in Figure 5, the pin angle number of the connector can be seen from one side of the component, and then rotate it 180 ° , which is the part needs to be welded;

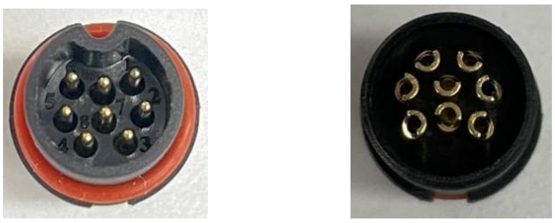


Figure 5

4. Use the AWG 26 cables to weld according to the pin angle definition in the welding manual (see appendix III for details). After the welding is completed, as shown in Figure 6;

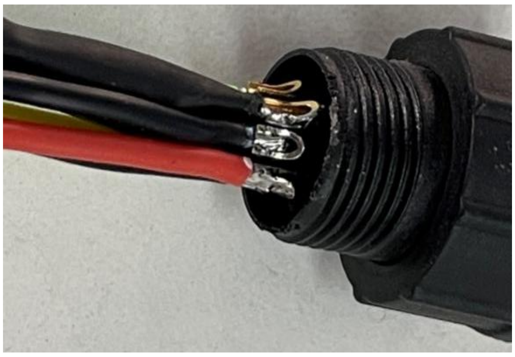


Figure 6

5. Take out the two parts shown in Figure 7 and put them on the welded parts, as shown in Figure 8；

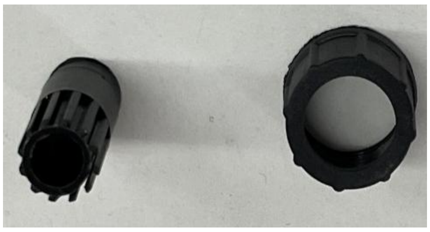


Figure 7

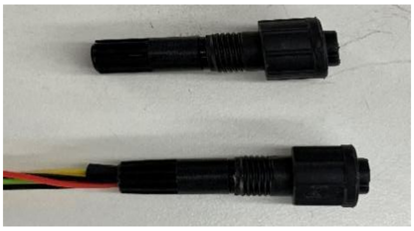


Figure 8

6. Take out the part shown in Figure 9, put them on the previously assembled parts, and tighten them, as shown in Figure 10；

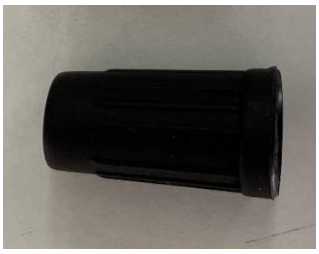


Figure 9

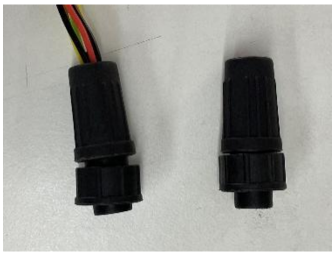


Figure 10

7. Then connect the remote control receiver and serial port, as shown in Figure 11；

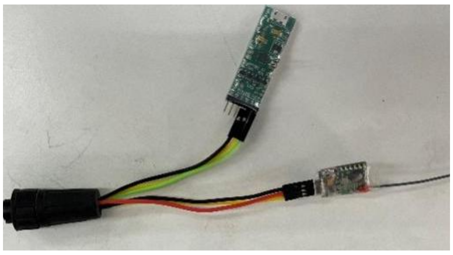


Figure 11

8. The welding method of 2pin connector is the same as that of 8pin connector.

Appendix III Connector pin angle definition instructions

Connector	Pin number	Define	Wire size	Remark	Color
8pin	1	CANH	AWG26	CAN	Red
	2	CANL	AWG26		Gray
	3	TX	AWG26	Serial port	Blue
	4	RX	AWG26		Green
	5	GND	AWG26		White
	6	5V	AWG26	Remote control receiver	Brown
	7	GND	AWG26		Black
	8	S.B PPM	AWG26		Yellow
2pin	1	Power+	AWG16	power supply for upper system	Red
	2	Power-	AWG16		Black

Appendix IV C/C++ APIReference documents

int Init_Comcore(void)

Function: initialization of host computer dynamic link library

Parameter: none

Return value: 0: initialization succeed;

Other: initialization fail

void exit_Comcore(void)

Function: exit initialization of host computer dynamic link library

Parameter: none

Return value: none

void aprctrl_datastamped_jni_register(saprctrldatastampedt* f)

Function: registration via the callback function provided by parameter, and this callback function conducts the sensor data processing.

Parameter: f is a struct pointer, and this struct includes the unique function pointer member variables.

Return value: none

void aprctrl_*eventcallback*_jni_*register(s*aprctrl*event*t* f)

Function: registration via the callback function provided by parameter, and this callback function conducts the processing of event code.

Parameter: f is a struct pointer, and this struct includes the unique function pointer member variables.

Return value: none

uint16*t get*_err_*state(board*name*e board*name)

Function: acquire the software/firmware runtime error code

Parameter: board name refers to the software/firmware ID

Parameter is one of the following values:

Host computer ID

Motor board ID

Central board ID

BMS ID

Return value: error code

int16*t get*_bat_soc(void)

Function: acquire percentage of battery remaining capacity

Parameter: none

Return value: percentage of battery remaining capacity (1~100)

int16*t get*_bat_charging(void)

Function: inquire whether the battery is in charging state

Parameter: none

Return value: 0: not in charging state

1: in charging state

int16*t get*_bat_mvol(void)

Function: acquire real-time voltage of battery

Parameter: none

Return value: voltage value, unit mV

int16*t get*_bat_mcurrent(void)

Function: acquire real-time current of battery

Parameter: none

Return value: current value, unit mA

int16*t get*_bat_temp(void)

Function: acquire battery temperature

Parameter: none

Return value: temperature value, unit degree Celsius

int16*t get*_chassis_*work*_model(void)

Function: acquire working state of chassis motor

Parameter: none

Return value: 1: motor in augmentation;

0: motor not in augmentation

int16*t get*_chassis_*load*_state(void)

Function: acquire setting value of chassis based on controlling parameter of different loading

Parameter: none

Return value:

0: no-load control parameter;

1: full load controlling parameter

int16*t get*_chassis_mode(void)

Function: acquire working mode of chassis finite state machine (FSM)

Parameter: none

Return value: 0 locking mode;

1 vehicle control mode;

2 pushing mode;

3 emergency stop mode;

4 error mode

int16*t get*_ctrl_*cmd*_src(void)

Function: acquire command origin of motor chassis control

Parameter: none

Return value:

0: control vehicle with remote controller;

1: control vehicle with host computer

int16*t get*_vehicle_meter(void)

Function: acquire the mileage since the chassis is power up

Parameter: none

Return value: mileage value, unit meter

uint16*t get*_host_version(void)

Function: acquire the host computer software version

Parameter: none

Return value: host computer software version number

uint16*t get_chassis_central_version*(void)

Function: acquire the central board firmware version

Parameter: none

Return value: the central board firmware version number

uint16*t get_chassis_motor_version*(void)

Function: acquire the motor board firmware version

Parameter: none

Return value: the motor board firmware version number

int16_t   get_line_forward_max_vel_fb (void)

Function:　acquire the forward speed limiting feedback value of the chassis

Parameter:　None

Return value:　the forward speed limiting feedback value of the chassis

int16_t　get_line_backward_max_vel_fb (void)

Function:　acquire the backward speed limiting feedback value of the chassis

Parameter:　None

Return value: the backward speed limiting feedback value of the chassis

int16_t　get_angular_max_vel_fb (void)

Function:　acquire the angular speed limiting feedback value of the chassis

Parameter:　None

Return value: the angular speed limiting feedback value of the chassis

int16_t getIapTotalProgress (void)

Function: Get the progress of IAP upgrades

Parameter: None

Return value:

−1: IAP upgrade failed

0: IAP upgrades are idle or started or interrupted

100: IAP upgrade completed

Other: Percentage of IAP upgrade progress

void iapCentralBoard (void)

Function: IAP upgrade of the central board firmware of the chassis

Parameter: None

Return value: none

Note: You need to place the central board firmware "central.bin" in the path of "/sdcard/firmware/" in advance.

void iapMotorBoard (void)

Function: IAP upgrade of the motor board firmware of the chassis

Parameter: None

Return value: none

Note: You need to place the motor board firmware "motor.bin" in the path of "/sdcard/firmware/" in advance.

bool isHostIapOver (void)

Function: Query if the IAP upgrade process has ended

Parameter: None

Return value: true: the IAP completes or fails or is interrupted

False: IAP not started or in progress

Int16_t getHostIapResult (void)

Function: acquire the reason for the end of IAP

Parameter: None

Return value:

3: IAP completes

4: IAP fails

5: IAP is interrupted

Others: IAP not started or in progress

Int16_t getHostIapErrorCode (void)

Function: Gets the error code for IAP failure

Parameter: None

Return value: the error code for IAP failure

int16_t get_chassis_hang_mode(void)

Function: Gets the setting state of chassis hang_mode

Parameter: None

Return value:

1: The chassis is in hang_mode

0: The chassis is not in hang_mode

int16_t get_charge_mos_ctrl_status (void)

Function: Gets the status of switch for charging MOS on the central board.

Parameter: None

Return value:

1: The MOS opened

0: The MOS closed

void set_*cmd*_vel(double linear*x,double angular*z)

Function: set up the command value of chassis target speed, which needs to be regular transmit once the chassis is enabled. It will be determined as communication failure if the chassis can’t receive the command value in continuous 150ms in controlling mode.

Parameter: linear_x: linear velocity command value, unit m/s;

angular_z: angular velocity command value, unit rad/s

Return value: none

void set_*line*_forward_*max*_vel(double linear*forward*max_x)

Function: set up the max forward linear velocity value of chassis.

Parameter: linearforwardmax_x: max forward linear velocity value of chassis, unit m/s, range 0−3

Return value: none

void set_*line*_backward_*max*_vel(double linear*backward*max_x)

Function: set up the max backward linear velocity value of chassis.

Parameter: linearbackwardmax_x: max backward linear velocity value of chassis, unit m/s, range −2−0

Return value: none

void set_*angular*_max_*vel(double angular*max_z)

Function: set up the max angular velocity command value of chassis.

Parameter: angularmaxz: the max angular velocity command value, unit rad/s, range 0−3

Return value: none

void set_*enable*_ctrl(uint16*t enable*flag)

Function: set up to enable the chassis to control the vehicle.

Parameter: enable_flag:

1 enable the vehicle control;

0 exit the vehicle control

Return value: none

void set_*smart*_car_*serial(const char * serial*no)

Function: set up the terminal name of serial port of host computer, e.g. ttyUSB0.

Parameter: serial_no: terminal name of serial port, under the path /dev/ by default, e.g. “ttyUSB0”

Return value: none

void set_comu_interface *(comu_choice_e comu_choice*)

Function: Set up the communication interface between the host computer and the chassis, including serial communication and CAN communication

Parameter: *comu_choice*:

‘comu_serial’ Use a serial port for communication

‘comu_can’ Use a CAN port for communication

Return value: none

void set_*chassis*_load_*state(int16*t newLoadSet)

Function: set up the parameter of chassis control based on the different chassis load.

Parameter: newLoadSet:

0: no-load parameter;

1: full load parameter

Return value: none

void set_chassis_poweroff (void)

Function: chassis power off controlled by host computer.

Parameter: none

Return value: none

void set_*remove*_push_cmd(void)

Function: when the chassis is in the pushing mode, it can exit this mode under the control of host computer.

Parameter: none

Return value: none

void setHostIapCanceled (void)

Function: Interrupt the IAP upgrade process.

Parameter: none

Return value: none

void set_chassis_hang_mode(int16_t enterHand)

Function:　The chassis is configured to be in hang test mode. When in hang_ode, the chassis can be controlled normally.

Parameter :　enterHand：

1：　Config the chassis to be in hang_mode;

0：　Config the chassis to be in non hang_mode.

Return value: none

void set_charge_mos_ctrl (bool on)

Function:　Sets the switch for charging MOS on the central board.

Parameter :　enterHand：　1：　Turn on the charging MOS switch;

0：　Turn off the charging MOS switch.

Return value: none