SEGWAY ROBOTICS

# User Manual

Panda

# Safety

Improper use of the PANDA can cause personal injury, death and/or property damage from loss of control, collision, and falls. To reduce risk of injury, read and follow all instructions and warnings in this manual.

The following safety messaging conventions are used throughout this document:

| | |
|---|---|
| WARNING! | Warns you about actions that could result in death or serious injury. |
| CAUTION! | Warns you about actions that could result in minor or moderate injury. |
| NOTICE | Indicates information considered important, but not related to personal injury. |

## WARNING!

- Do not sit, stand or ride on PANDA. It may cause injury.
- Do not control PANDA to hit people or animals. Collision may cause injury.
- When PANDA is running, it needs to remind people nearby at all times. Accidental collisions with PANDA may result in injuries.
- PANDA can be accelerated quickly, and customers are advised to practice at a low speed until users are familiar with controlling PANDA.

The unexpected movement of PANDA may cause injuries.

- Do not attempt to disassemble the battery, which may result in electric shock, burns or even fire. Trying to open the battery case can damage the battery case and release toxic substances. It can also render the battery unusable.
- As with all rechargeable batteries, do not charge near flammable materials, which could cause a fire.
- If the battery shell is damaged or the battery emits peculiar smell, smoke, overheat, or leaks, do not continue to use the battery. Do not touch any substance that oozes from the battery, which may cause poisoning.
- Strictly observe and follow all safety information on the warning label on the battery. Failure to do so can result in injury or even death.
- Do not use badly worn or damaged cables, which may shock yourself or damage PANDA.

## CAUTION!

- Set performance parameters correctly and carefully. PANDA follows the commands issued to it, and it is the user's responsibility to implement correct and safe performance parameters.
- Not charging the battery can cause permanent damage to the

battery.

- The battery can only be charged with the charger of PANDA.
- Before operating PANDA, please be sure to read the user manual and be familiar with the operation of PANDA and various precautions.

## NOTICE

- In the absence of communication with our company, our company will not take any responsibility for any accident caused by the modification of the chassis.

# Index

# 1 Product introduction

## 1.1 Product schematic diagram

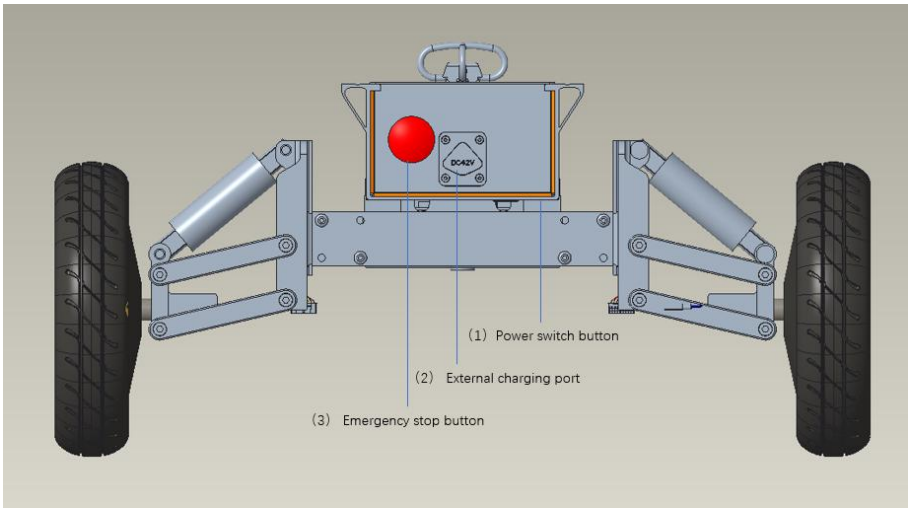


Figure 1

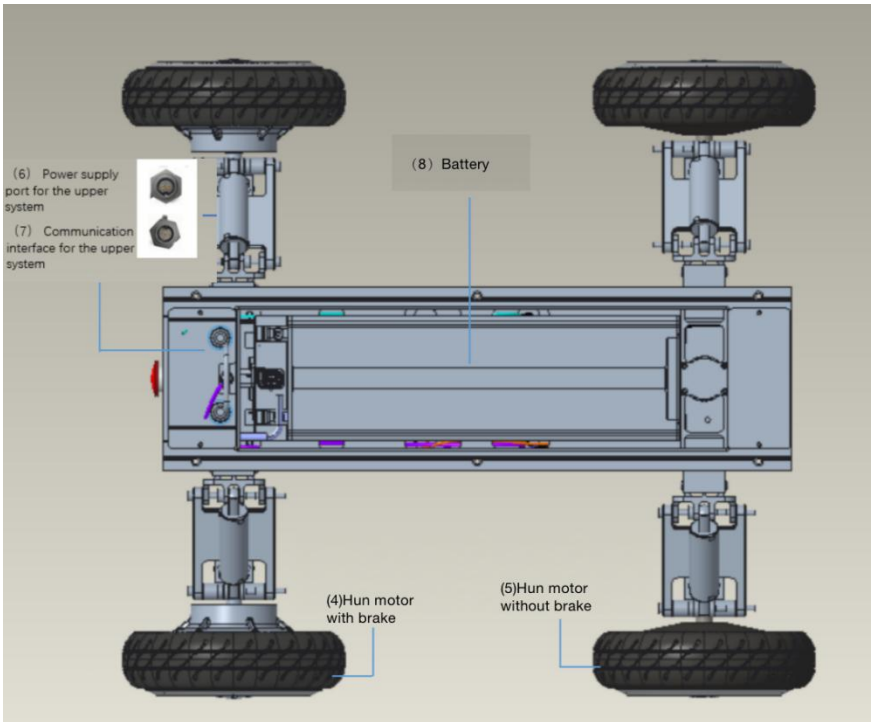


Figure 2

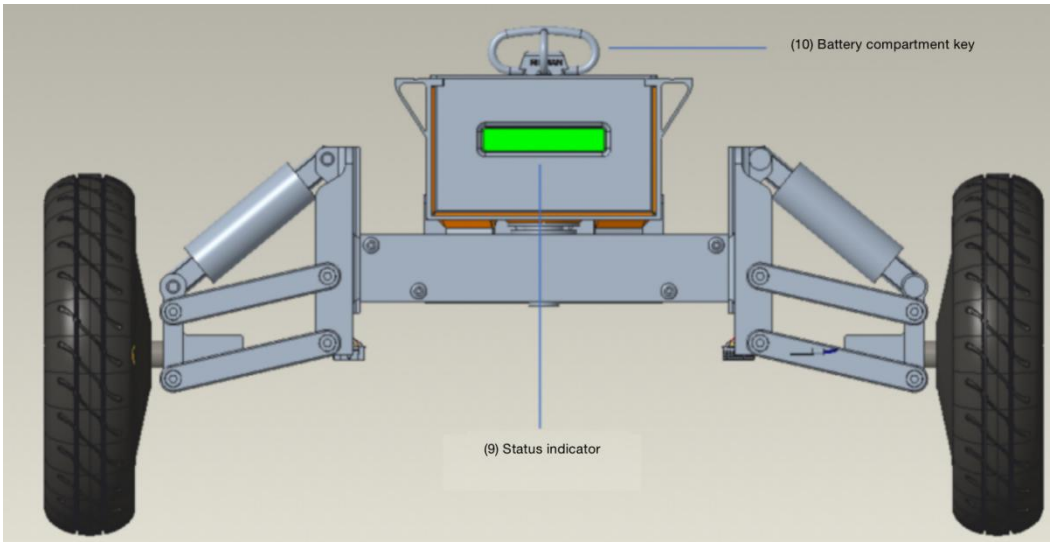


Figure 3

## 1.2 Component Introduction

Table 1

| Number | Component name | Description |
|---|---|---|
| 1 | Power switch button | Start up: long press the power switch button, until the indicator light is steady on and accompanied by a prompt tone, the chassis starts up successfully, the chassis is in lock mode, and the indicator light is steady yellow<br>Shutdown: long press the power switch button until the prompt sound starts, release the power switch button, the chassis shuts down successfully, at this time, the power switch button lamp and indicator light are all off. |
| 2 | External charging port | Connect the charger to charge the device |
| 3 | Emergency stop button | Used to switch the chassis to emergency stop mode in emergency state |
| 4 | Hub motor with brake | 8 inch tire, with power-off brake |
| 5 | Hub motor without brake | 8 inch tire |
| 6 | Power supply port for the upper system | Supplies power to the upper system |
| 7 | Communication interface for the upper system | Includes CAN, serial port, and remote control receiver |
| 8 | Battery | Used to power the entire system |
| 9 | Status indicator | Indicator color and status represent different modes of the product |
| 10 | Battery compartment key | It opens the lock that holds the battery |

# 1.3 The remote control

## 1.3.1 Schematic diagram of remote control


Figure 4

* The forward and reverse input (throttle or direction) of the remote control can be achieved by flipping the phase switch under the T8FB.


Figure 5

*Remote control alarm voltage adaptive 2S, 3S, 4S lithium batteries and 4 nickel metal hydride batteries, that is, if T8FB is powered by 2S, 3S, 4S lithium batteries or 4 nickel metal hydride batteries, after connecting the batteries, T8FB will automatically set the low voltage alarm value according to the battery type.

### 1.3.2 Receiver pair

Each transmitter has its own ID code. Before starting to use the device, the receiver must pair with the transmitter. After pairing, the ID code is stored in the receiver and does not need to be paired again unless the receiver is paired with another transmitter. When you buy a new receiver, you must re-code it, otherwise the receiver will not work properly.

(1) Place the remote control and receiver horizontally, and the distance between them is about 50cm;

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

(3) Press the ID SET key on the side of the receiver for more than 1 second, and the LED light of the receiver starts flashing quickly, indicating that the code is being matched, and the receiver will look for the nearest remote control for matching the code;

(4) Stop flashing the LED light of the receiver, which means that the code is completed. If the LED light of the receiver blinks slowly, it means that the code fails, and the code needs to be repeated.

### 1.3.3 Remote control instructions

（1）PANDA chassis startup: press the PANDA power switch button;

Note: Please check the PANDA status. Hold down the power switch

button until the buzzer does not beep continuously and the indicator 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 emergency stop state and enter the enable state. That is, the emergency stop switch is not under, and the enable switch is shifted from the top to the bottom.

（3）PANDA is in Normal mode at this time. See the following table for detailed operations:

Table 2 Remote control operation

| Control action | Remote control operation |
|---|---|
| Turn right or left | Turn the rudder lever left and right |
| Move forward or back | The throttle lever moves forward and backward |
| Emergency stop/exit emergency stop | Remote control emergency stop switch:<br>Top - exit emergency stop;<br>Bottom - Open emergency stop |
| Adjust the maximum angular velocity | Adjustment knob for maximum angular velocity of remote control. If you turn left, the maximum angular velocity decreases. If you turn right, the maximum angular velocity increases. |
| Adjust the maximum linear velocity | Adjustment knob for maximum linear speed of remote controller. If you turn left, the maximum linear velocity decreases. If you turn right, the maximum linear velocity increases. |
| Enable/disable the function | Turn the enable switch from the top to the bottom to enable chassis.<br>Turn the enable switch from the bottom to the top to disable the chassis. |

### 1.3.4 The upper system controls the car explanation

The upper system is a PC terminal control computer, which can directly issue control commands, and various information changes are

displayed on the screen. The upper system controls the chassis and provides some necessary operating environment for the chassis, and extends the man-machine control or demonstration functions provided by the chassis. The upper system has the characteristics of leading management, coordinating resources, monitoring agent and controlling PANDA.

（1）PANDA chassis startup: press the PANDA power switch button;

Note one: Please check the PANDA status. Hold down the power switch button until the buzzer does not beep continuously and the indicator is steady yellow.

Note two: The remote control cannot be turned on when the upper system controls the car. Or after the remote control is turned on, turn the enable switch to the top.

（2）Ensure that the PANDA serial cable or CAN line is connected to the upper system;

（3）Grant permission to the /sdcard/segway/hardware_log/ folder. Otherwise, the new log file will fail. Give /catkin_ws/SRC/RosCode segwayPanda/lib/ all files and directory add permissions (after the first setting, you do not need to reset it) :

‘’

`cd /sdcard/segway/hardware_log`

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

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

`sudo chmod 777 *

‘’

（4）According to the file of txt under ‘catkin_ws/src/RosCode /segwayPanda/Cmakelists.txt’, choose the compile option based on the upper system platform (x86_64 or arm). The sample below is complied based on x86_64 platform. Disable ‘libctrl_arm64-v8a.so’ by using ‘#’ (after the first setting, you do not need to reset it) :

‘’

`target_link_libraries(SmartCar`

`${catkin_LIBRARIES}`

`#${PROJECT_SOURCE_DIR}/lib/libctrl_arm64-v8a.so

//under x86_64 platform, enable the line, disable the line under ARM`

`${PROJECT_SOURCE_DIR}/lib/libctrl_x86_64.so

// under arm platform, enable the line, disable the line under x86_64`

‘’

（5）Enter the ROS workspace and run the following command to compile the Segway_MSgs package message.

‘’

cd catkin_ws

```
catkin_make
-DCATKIN_WHITELIST_PACKAGES='segway_msgs'
```

（6）Enter the ROS workspace and run the following command to compile the Segwayrmp package.

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

（7）Execution of vehicle control in ROS：

1）To create a terminal, run the following command

```
cd catkin_ws
roscore
```

2）Create a new terminal and run the SmarCar node

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

3）Create a new terminal and run the following command to run the routine test node

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

# 2 Software introduction

This chapter describes the related files, software interface functions, and fault code information provided by PANDA.

## 2.1 The file provided to the user

Table 3 Files provided

| Files | Function |
| --- | --- |
| Libctrl_x86_64.so | Provides the x86 platform C/C++ chassis related interface |
| Libctrl_arm64-v8a.so | Provides arm platform C/C++ chassis related interface |
| Comm_ctrl_navigation.h | The C/C++ API interface head files |
| ROS package | Provides ROS nodes for chassis control |

## 2.2 Interface function introduction

### 2.2.1 C/C++interface introduction

Table 4 callback data type

| The callback type | Index of the callback | Function description | Data structure |
|---|---|---|---|
| Chassis_Data_Motors_Speed | 1 | Chassis 4 wheel speed information | typedef struct{<br>int16_t fl_speed;<br>int16_t fr_speed;<br>int16_t rl_speed;<br>int16_t rr_speed;<br>}chassis_motors_speed_data_t;<br> |
| **Chassis_Data_Car_Speed** | 2 | Get chassis speed information | typedef struct{<br>int16_t car_speed;<br>int16_t turn_speed;<br>}chassis_car_speed_data_t; |
| Chassis_Data_Front_Ticks | 3 | Chassis front two wheel encoder information | typedef struct{<br>int32_t fl_ticks;<br>int32_t fr_ticks;<br>}front_motors_ticks_t; |
| **Chassis_Data_Rear_Ticks** | 4 | Chassis rear two wheel encoder information | typedef struct{<br>int32_t rl_ticks;<br>int32_t rr_ticks;<br>}rear_motors_ticks_t; |
| Chassis_Data_Odom_Pose_xy | 5 | Odom pose information | typedef struct{<br>float pos_x;<br>float pos_y;<br>}odom_pos_xy_t; |
| Chassis_Data_Odom_Euler_xy | 6 | Odom Euler X/Y information | typedef struct{<br>float euler_x;<br>float euler_y;<br>}odom_euler_xy_t; |
| Chassis_Data_Odom_Euler_z | 7 | Odom Euler Z information | typedef struct{<br>float euler_z;<br>}odom_euler_z_t; |
| Chassis_Data_Odom_Linevel_xy | 8 | Odom speed X/Y information | typedef struct{<br>float vel_line_x; |

| | | | |
|---|---|---|---|
| | | | float vel_line_y;<br>}odom_vel_line_xy_t; |
| Chassis_Data_Imu<br>_Gyr | 9 | Gyroscope data | typedef struct{<br>int16_t gyr[3];<br>}imu_gyr_original_data_; |
| Chassis_Data_Imu<br>_Acc | 10 | Accelerometer data | typedef struct{<br>int16_t acc[3];<br>}imu_acc_original_data_; |

Note 1: Odom data: the default heading Angle at startup is 0 degrees.

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

Table 5 event definition

| Event type | Index of event | Function description |
|---|---|---|
| ChassisBootReadyEvent | 1 | The chassis center panel is started |
| PadPowerOffEvent | 2 | The chassis to turn it off |
| OnEmergeStopEvent | 3 | Enter the emergency stop |
| OutEmergeStopEvent | 4 | Exit the emergency stop |
| OnLockedRotorProtectEvent | 5 | The inability of a wheel to rotate due to an external force |
| OutLockedRotorProtectEvent | 6 | The failure of the wheel to rotate due to external forces was eliminated |
| OnLostCtrlProtectEvent | 7 | The wheel appears to rotate rapidly without control |
| OutLostCtrlProtectEvent | 8 | The rapid rotation phenomenon is eliminated |
| CalibrateGyroSuccess | 9 | Gyroscope calibration successful |
| CalibrateGyroFail | 10 | Failed to calibrate the gyroscope |
| CalibratePasheCurrentSuccess | 11 | Calibration of phase current succeeded |
| CalibratePasheCurrentFail | 12 | Failed to calibrate phase current |
| ChassisLockRotorWarning | 13 | Locked-rotor occor and |

| | | |
|---|---|---|
| | | then warning |

Table 6 get/set interface

| Interface name | Interface description |
|---|---|
| get_err_state | Get error code for upper system/central board/motor board/battery |
| get_bat_soc | Obtain the percentage of the remaining battery power |
| get_bat_charging | Obtain the battery charging status (1: charging; 0: non-charging) |
| get_bat_mvol | Obtain battery voltage (unit: millivolt) |
| get_bat_mcurrent | Obtain battery current (unit: milliampere) |
| get_bat_temp | Obtain the battery temperature (in Degrees Celsius) |
| get_chassis_work_model | Get chassis working state (0: wheels have no power; 1: the wheel have power) |
| get_chassis_load_state | Get chassis load parameter Settings (0: no load; 1: full load) |
| get_chassis_mode | Get the chassis state machine (0: lock the car; 1: car control; 2. Implementation; 3: emergency stop; 4: False) |
| get_ctrl_cmd_src | Get the chassis current control source (0: remote control; 1: host computer) |
| get_vehicle_meter | Get chassis mileage (unit meters) |
| get_host_version | Obtain the version number of the upper computer |
| get_chassis_central_version | Obtains the version number of the central board |
| get_chassis_motor_version | Get motor board version number (reserved) |
| get_line_forward_max_vel_fb | Obtain chassis forward speed limit value (unit meters per hour) |
| get_line_backward_max_vel_fb | Get chassis backward speed limit value (unit meters per hour) |
| get_angular_max_vel_fb | Get the chassis angular velocity limit (milliradians per second) |
| getIapProgress | Get IAP progress |
| iapCentralBoard | IAP upgrades to the central board |
| iapMotorBoard | IAP upgrades to the motor board |
| iapBrakeBoard | IAP upgrades to the brake board(with the brake board) |
| isHostIapOver | Check whether IAP ends |
| getHostIapResult | Get IAP results (3: complete; 4: failure. 5: Interruption. 0: meaningless) |
| getHostIapErrorCode | Get the IAP error code |
| get_calibrate_mid_value_status | Check whether the front wheel median has been corrected (1: corrected; 0: uncorrected) |
| set_cmd_vel | Set chassis linear velocity and angular velocity (unit meters per second and radians per second) |
| set_line_forward_max_vel | Set chassis forward speed limit (in meters per second) |
| set_line_backward_max_vel | Set chassis backward speed limit (in meters per second) |

| | |
|---|---|
| set_angular_max_vel | Set chassis angular velocity limit (unit radians per second) |
| set_enable_ctrl | Set the enable state of the chassis upper system controlled vehicle (1: Enable; 0: Disable) |
| init_control_ctrl | Chassis initialization interface |
| exit_control_ctrl | Chassis exit initialization interface |
| set_smart_car_serial | Set the name of the serial port used by the dynamic library of the upper computer |
| set_comu_interface | Set the communication interface with the chassis (0: serial port; 1: CAN) |
| set_chassis_load_state | Set chassis load parameters (0: no load; 1: full load) |
| set_chassis_poweroff | Issue the chassis shutdown command |
| setHostIapCanceled | Cancel IAP command on upper computer |
| set_calibrate_mid_value | The command to correct the median Angle of the first two rounds is delivered |
| reset_host_power_time_s | Issue a command to reset the upper mechanical power of the chassis (unit: second; Maximum interval: 65535 seconds) |
| clear_chassis_error_code | **Clear chassis error codes, excluding alarms, exceptions, and battery errors. Use with caution** |
| enable_rotate_switch | **Enable the chassis in-situ rotation function** |

## 2.2.2 ROS interface introduction —SmartCar

Table 7 News release

| Topic Name | Function decription | Message Type | Message Type Info | Frequency |
|---|---|---|---|---|
| Bms_fb | Battery Information | Segway_msgs/ Bms_fb | int16 bat_soc<br>int16 bat_charging<br>int32 bat_vol<br>int32 bat_current<br>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 state machine | Segway_msgs/ Chassis_mode_fb | uint16 chassis_mode | 1 |
| Error_code_fb | Chassis error code | Segway_msgs/ Error_code_fb | uint32 host_error<br>uint32 central_error<br>uint32 front_left_motor_error<br>uint32 front_right_motor_error<br>uint32 rear_left_motor_error<br>uint32 rear_right_motor_error<br>uint32 bms_error | 1 |
| Motor_work_mode_fb | Chassis working condition | Segway_msgs/ Motor_work_mode_fb | uint16 motor_work_mode<br>#0: no output torque 1: output torque | 1 |
| Speed_fb | Chassis speed | Segway_msgs/ Speed_fb | float32 car_speed<br>float32 turn_speed<br>float32 fl_speed<br>float32 fr_speed<br>float32 rl_speed<br>float32 rr_speed | 40 |

| | | | | |
|---|---|---|---|---|
| | | | uint64<br>speed_timestamp | |
| Ticks_fb | Chassis encoder information | Segway_msgs/ Ticks_fb | int32 fl_ticks<br>int32 fr_ticks<br>int32 rl_ticks<br>int32 rr_ticks<br>uint64 ticks_timestamp | 40 |
| Odom | Odom data | Nav_msgs/odom | | 40 |
| Imu | Imu data | Sensor_msgs/imu | | 40 |

Table 8 News subscription

| TopicName | Function decription | Message Type | Message Type Info |
|---|---|---|---|
| Cmd_vel | Control chassis movement | Geometry_msgs/twist | Angular.z //rad/s<br>Linear.x //m/s |

Table 9 service client

| Service name | Function decription | Message type | Message type info |
|---|---|---|---|
| chassis_send_event_srv | Send event number | Segway_msgs/chassis_send_event | chassis_send_event_id--ros_is_received |

Table10 service server

| Service name | Function decription | Message type | Message type info |
|---|---|---|---|
| ros_clear_chassis_error_code_cmd.srv | **Clear chassis error codes, excluding alarms, exceptions, and battery errors. Use with caution** | Segway_msgs/ ros_clear_chassis_error_code_cmd | bool clear_chassis_error_code_cmd<br>---<br>uint8 clear_chassis_error_code_result |
| ros_enable_chassis_rotate_cmd.srv | **Enable the chassis in-situ** | Segway_msgs/ ros_enable_chassis_rotate_cmd | bool ros_enable_chassis_rotate_cmd |

| | | | |
|---|---|---|---|
| | rotation function | | ---<br>int16 chassis_enable_rotate_result |
| ros_get_chassis_rotate_switch_cmd.srv | Query whether the chassis is rotating in place | Segway_msgs/ros_get_chassis_rotate_switch_cmd | bool ros_get_chassis_rotate_cmd<br>---<br>uint8 chassis_rotate_state |
| ros_get_chassis_SN_cmd.srv | Get the SN of the chassis central control MCU | Segway_msgs/ros_get_chassis_SN_cmd | bool ros_get_chassis_SN<br>---<br>string chassis_SN |
| ros_get_load_param_cmd_srv | Getting load Settings | Segway_msgs/ros_get_load_param_cmd | ros_get_load_param<br>---<br>get_load_param #0:no_load, 1: full_load |
| ros_get_sw_version_cmd_srv | Obtaining the Software Version | Segway_msgs/ros_get_sw_version_cmd | ros_get_sw_version_cmd<br>---<br>uint16 host_version<br>uint16 central_version<br>uint16 motor_version |
| ros_get_vel_max_feedback_cmd_srv | Get the speed limiter | Segway_msgs/ros_get_vel_max_feedback_cmd | ros_get_vel_max_fb_cmd<br>---<br>forward_max_vel_fb<br>backward_max_vel_fb<br>angular_max_vel_fb |
| ros_set_chassis_enable_cmd_srv | The chassis enable command is issued | Segway_msgs/ros_set_chassis_enable_cmd | ros_set_chassis_enable_cmd<br>---<br>chassis_set_chassis_enable_result |
| ros_set_chassis_poweroff_cmd_srv | Issue the chassis shutdown command | Segway_msgs/ros_set_chassis_poweroff_cmd | ros_set_chassis_poweroff_cmd<br>---<br>chassis_set_poweroff_result |
| ros_set_load_param_cmd_srv | Setting chassis load | Segway_msgs/ros_set_load_param_cmd | ros_set_load_param<br>#0:no_load, 1: full_load<br>---<br>chassis_set_load_param_result |
| ros_set_vel_max_cmd_srv | Set the speed limit | Segway_msgs/ros_set_vel_max_cmd_srv | ros_set_forward_max_vel<br>ros_set_backward_max_vel<br>ros_set_angular_max_vel<br>--- |

| | | | |
|---|---|---|---|
| | | | chassis_set_max_vel_result |
| ros_reset_host_power_cmd_srv | Chassis reset upper power | segway_msgs/ros_reset_host_power_cmd | uint16 reset_interval_time<br>---<br>uint8 reset_result |

Table11 action server

| Action name | Function decription | Message type | Message type info |
|---|---|---|---|
| ros_set_iap_cmd_action | IAP upgrade of board firmware | Segway_msgs/ros_set_iap_cmdAction | uint16 board_index_for_iap<br>---<br>Int16 iap_result<br>#3: iap_state_complete; 4: iap_state_fail; 5: iap_state_abort<br>Int16 error_code<br>#When iap_result value is 4, this value represents the error code<br>---<br>Int16 iap_percent |

### 2.2.3 Fault code information table

The fault code is obtained through the uint32_get_err_state (board_name_e board_name) interface. The corresponding information is as follows:（**Note: "Manual Force Clear Errors" needs to be implemented very carefully**）：

Table12 Fault code

| Board name | Bit | Error info | 底盘动作 | 处理 |
|---|---|---|---|---|
| host | 0x00000000 | No error | | |
| | 0x00000001 | The central board is out of contact | Level 2, unable to control the car | Check communication |
| | 0x00000002 | The serial port module is | Level 2, unable to | Plug back the module |

| | | | | |
|---|---|---|---|---|
| | | removed | control the car | |
| Central | 0x00000000 | No error | | |
| | 0x00000001 | Vehicle control command communication is interrupted | Level 2, chassis parking, lock mode | Check communication |
| | 0x00000002 | Motor board communication is interrupted | Level 2, chassis parking, lock mode | Check motor board communication |
| | 0x00000004 | IMU initialization failed | Level 1, controllable car, wrong angle | Drive back at low speed, check hardware |
| | 0x00000008 | The IMU failed to read data | Level 1, controllable car, wrong angle | Drive back at low speed, check hardware |
| | 0x00000010 | The wheel whirled uncontrollably | Level 5, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000020 | The wheel cannot rotate because of external forces | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000040 | Failed to calibrate the IMU | Level 1, controllable car, wrong angle | Drive back at low speed, check hardware |
| | 0x00000080 | Flash read failure | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000100 | IMU data update failed | Level 1, controllable car, wrong angle | Drive back at low speed, check hardware |

| | | | | |
|---|---|---|---|---|
| | 0x00000400 | rollover | Level 2, Parking, Lock Mode | Make sure the chassis is not in a rollover state |
| | 0x00000800 | Restart of any motor board detected | Level 2, Parking, Lock Mode | Check motor board communication |
| | 0x00001000 | The front left wheel magnetic encoder is faulty | Level 5, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00002000 | The front right wheel magnetic encoder is faulty | Level 5, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00004000 | Battery communication is interrupted. | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00008000 | The rear left wheel magnetic encoder is faulty | Level 5, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00010000 | The rear right wheel magnetic encoder is faulty | Level 5, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00200000 | Abnormal front wheel angle convergence timeout | Level 2, Parking, Lock Mode | Righting the front wheel |
| Motor | 0x00000000 | No error | | |
| | 0x00000001 | Phase current fault | Level 4, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000002 | Phase voltage | Level 3, | Manually clear |

| | | | | |
|---|---|---|---|---|
| | | fault | Parking, Error Mode | errors, drive to safety at low speed, check hardware |
| | 0x00000004 | Lack of phase | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000008 | Voltage failure | Level 4, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000010 | Self-test failure | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000020 | Over current | Level 4, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000080 | The wheel cannot rotate because of external forces | Level 4, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000100 | Electrical Angle fault | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000200 | Overpower fault | Level 4, Parking, Error Mode | detect hardware |
| | 0x00000400 | Over speed fault | Level 5, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |

| | | | | |
|---|---|---|---|---|
| | 0x00000800 | Speed sensor fault | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| Battery | 0x00000000 | No error | No error | |
| | 0x00000200 | overshoot | Level 3, Parking, Error Mode | detect hardware |
| | 0x00000400 | The charging temperature exceeds the normal temperature | Level 3, Parking, Error Mode | detect hardware |
| BrakeSticking | 0x00000002 | Left brake open circuit | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000004 | Left brake short circuit | Level 4, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000008 | Left brake failed to close | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000200 | Right brake open circuit | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000400 | Right brake short circuit | Level 4, Parking, Error Mode | Manually clear errors, drive to safety at low speed, check hardware |
| | 0x00000800 | Right brake failed to close | Level 3, Parking, Error Mode | Manually clear errors, drive to safety at low |

| | | | | |
|---|---|---|---|---|
| | | | | speed, check hardware |
| | 0x00008000 | Brake board communication lost | Level 2, Parking, Error Mode | Detect brake board communication |

### 2.2.4 In-situ rotation function introduction

The in-situ rotation function is a function realized in the special mode of the chassis. When the linear velocity is 0 and the angular velocity is not 0, the chassis uses this function. When this function is used, the current of the rear wheel will be too large, which may cause abnormality of the chassis and the motor. It is recommended to use this function. Not necessary and not easy to use. After the chassis is powered on, it does not support the in-situ rotation function by default. If you really need to use this function, as described in Section 2.2.2, the host computer needs to use the "Enable chassis in-situ rotation mode" service of ROS to enable the chassis to support in-situ rotation again.

When using the chassis to rotate in place, if the chassis is stuck and cannot rotate and the rotor is blocked, the chassis will perform the following actions:

1. When the rotor is locked for about 5 seconds, the chassis cancels the support for the in-situ rotation function, and sends the locked rotor alarm event to the upper computer at the same time. At this time, it is necessary

to stop the rotation in place, and then let the chassis have a forward and backward movement component to release the electric energy accumulated by the chassis.

2. When the locked rotor occurs and the chassis still cannot release the electric energy for about 10 seconds, the chassis will automatically release the accumulated electric energy. At this time, it is necessary to prevent the chassis from being on a sloped ground.

3. When the locked rotor occurs, the hardware is damaged and the power cannot be released. After about 15 seconds, the chassis sends a locked rotor error event to the upper computer. At the same time, the chassis switches to the error mode, and the power is off and the brake is locked.

After the chassis releases power and returns to normal, if it is really necessary to continue to use the in-situ rotation function, as described in Section 2.2.2, the host computer needs to use the "Enable chassis in-situ rotation mode" service of ROS to enable the chassis to support in-situ again. rotation. At the same time, it should be noted that in order to protect the chassis, the above services cannot be used within 30 seconds after the chassis stops supporting the in-situ rotation function to enable the chassis to support the in-situ rotation function.

# 3 Firmware upgrade and version upgrade

IAP is a software function module of the system, that is, in application programming, that is, online to upgrade the MCU program. This function uses the upper computer to burn and write the new version bin file to the single chip microcomputer (including the central control board, motor driver board, etc.) when the program is running. The premise is that the single chip microcomputer bin file to be burned needs to be named according to the requirements of the upper computer, and then placed in the /sdcard/firmware/ path of the upper computer. Then the online upgrade is performed through commands on the terminal.

## 3.1 The firmware update

Before firmware upgrade, test the data communication between the upper computer and the lower computer to check whether the communication is normal. Test using commands at shell terminals.

（1）View the program path of the upper computer

Go to the path where the host program resides and check whether the executable file exists. Arm executable file, x86 executable file, ARM dynamic library and x86 dynamic library are shown below:

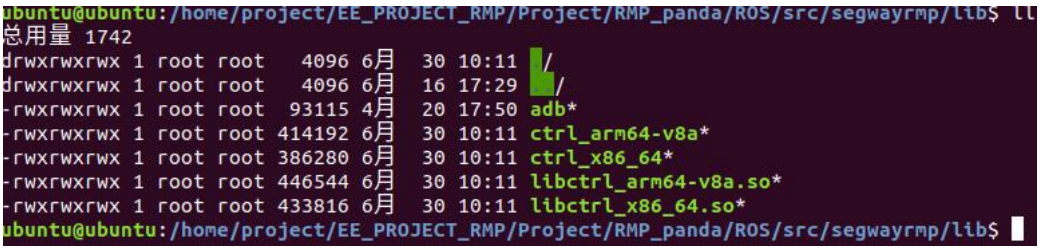


Figure 6

（2）View the software version of central board

Check the software version of the lower computer. This step can test the communication status of the upper computer and the lower computer at the same time. If you can check the software version of each plate of the lower computer through the upper computer, it indicates that the communication is good.

Center board test command：*./ctrl_x86_64 s -test central*

1） When connecting the serial port for the first time, if the USB port of the serial port is not granted with the execution permission, the program requires to obtain the root permission to modify the execution permission of the serial port USB port. In this case, you need to enter the system login password and press Enter, as shown in the following figure:


Figure 7

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

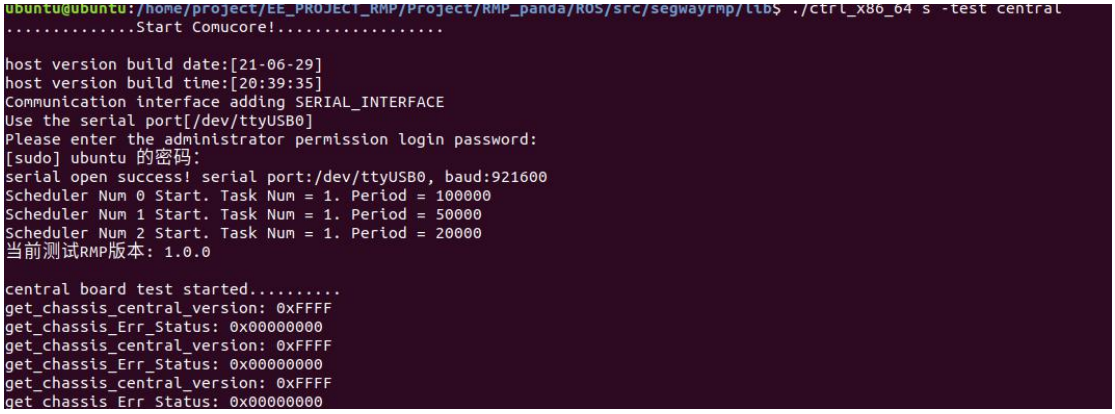


Figure 8

3）When the communication is successful, the version number is printed as follows, and the version number is non-0xFFFF. At this time, the communication between the upper computer and the single chip microcomputer is normal, and online upgrade can be carried out:

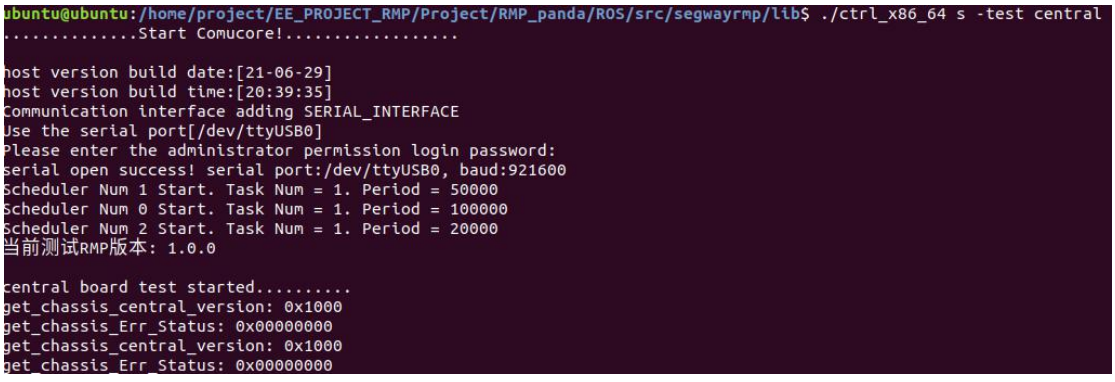


图 9

## 3.2 Version update

（1）Single-chip bin file placement

Put the bin file of the board software to be upgraded into the /sdcard/firmware path of the upper computer, central board software bin file ‘central.bin’, front wheel motor board bin file ‘motor_front.bin’, rear wheel motor board bin file ‘motor_rear.bin’ and brake board bin file ‘barke.bin’

```
ubuntu@ubuntu:/sdcard/firmware$ ll
总用量 272
drwxrwxrwx 2 root   root    4096 1月   7 10:33 ./
drwxrwxrwx 5 root   root    4096 7月  26 16:28 ../
-rwxrwxrwx 1 root   root   28708 11月 16 10:39 brake.bin*
-rwxrwxrwx 1 ubuntu ubuntu 91368 12月 31 14:59 central.bin*
-rwxrwxrwx 1 root   root   67788 11月 12 20:49 motor_front.bin*
-rwxrwxrwx 1 root   root   67788 11月 12 20:49 motor_rear.bin*
-rwxrwxrwx 1 root   root      40 10月 13 10:09 password.txt*
ubuntu@ubuntu:/sdcard/firmware$
```

Figure 10

（2）The bin file on the lower computer is written online

Enter the path of the program executable file ctrl_X86_64 or CTRl_ARM64-V8A on the upper computer, as follows:

```
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Project/RMP_panda/ROS/src/segwayrmp/lib$ ll
总用量 1763
drwxrwxrwx 1 root root   4096 6月  30 10:11 ./
drwxrwxrwx 1 root root   4096 6月  16 17:29 ../
-rwxrwxrwx 1 root root 114264 6月  30 14:13 adb*
-rwxrwxrwx 1 root root 414192 6月  30 10:11 ctrl_arm64-v8a*
-rwxrwxrwx 1 root root 386280 6月  30 10:11 ctrl_x86_64*
-rwxrwxrwx 1 root root 446544 6月  30 10:11 libctrl_arm64-v8a.so*
-rwxrwxrwx 1 root root 433816 6月  30 10:11 libctrl_x86_64.so*
ubuntu@ubuntu:/home/project/EE_PROJECT_RMP/Project/RMP_panda/ROS/src/segwayrmp/lib$
```

Figure 11

Run the following commands to upgrade each board online: Go to the program path of the upper computer and run the following

commands (use 's' when using a serial port; Use 'C' when using CAN port) :

Center board upgrade command:

*./ ctrl_x86_64 s –iap central*

Front wheel motor board upgrade command:

*./ ctrl_x86_64 s –iap motor_front*

Rear wheel motor board upgrade command :

*./ ctrl_x86_64 s –iap motor_rear*

Brake board upgrade command :

*./ ctrl_x86_64 s –iap brake*

For example, run the./ ctrl_x86_64 s -IAP central command to upgrade the central board, as shown in the following

figure:

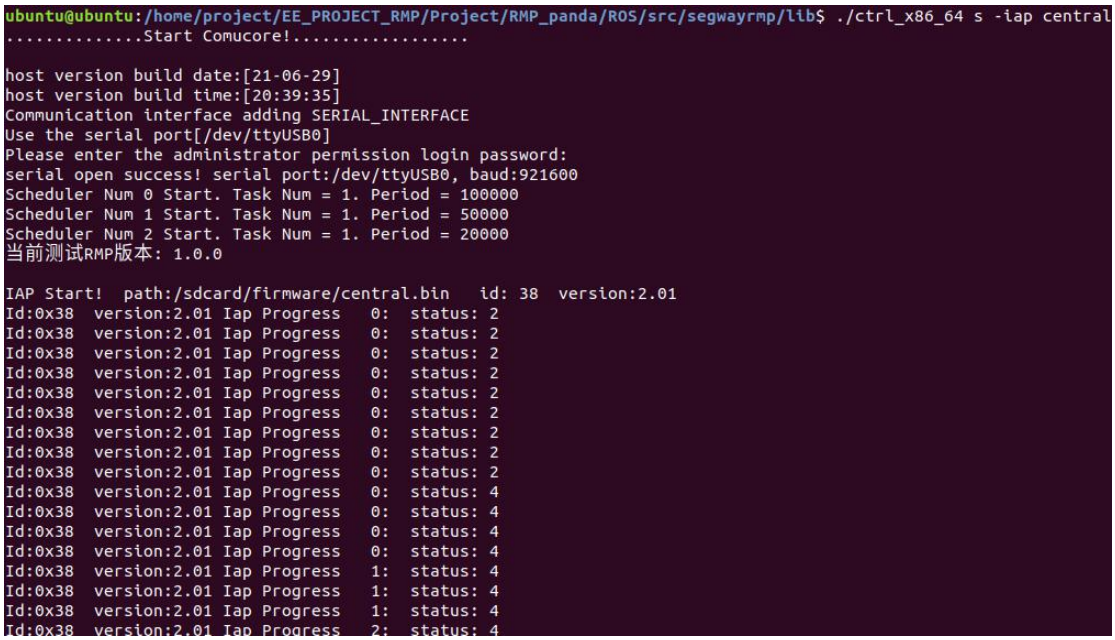


Figure 12

During the upgrade process, you can view the upgrade Progress. Progress indicates the percentage of IAP upgrade Progress. When the Progress value reaches 100, it indicates that the bin file of the routing board has been burned to the central board chip.

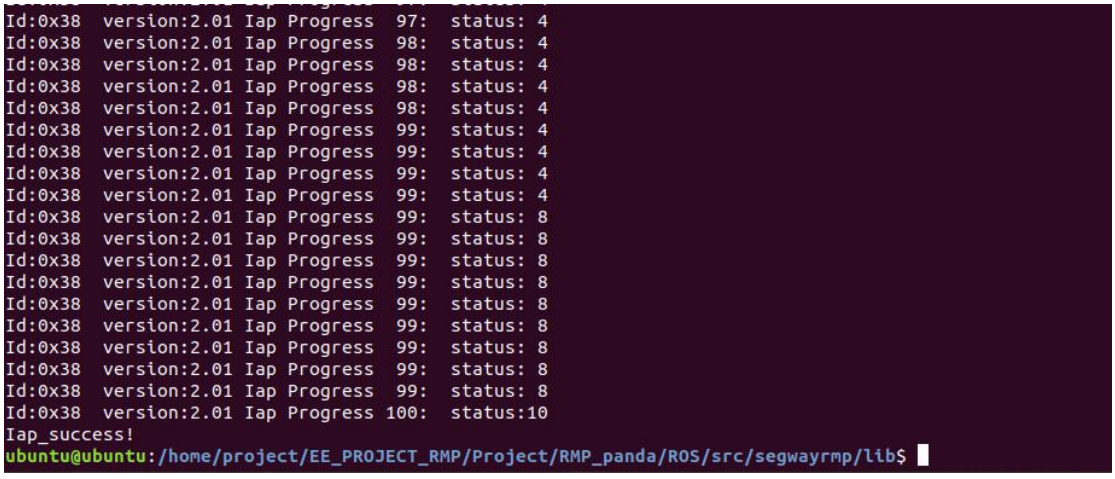


Figure 13

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

Run Step 1 to check the software version. In the path of the upper computer program, enter the command./ ctrl_x86_64 s -test central, as shown in the following figure:

In this case, the software version number of the central control board is 0x1000, indicating that the online upgrade is successful and the communication between the host computer and the central control board is good.

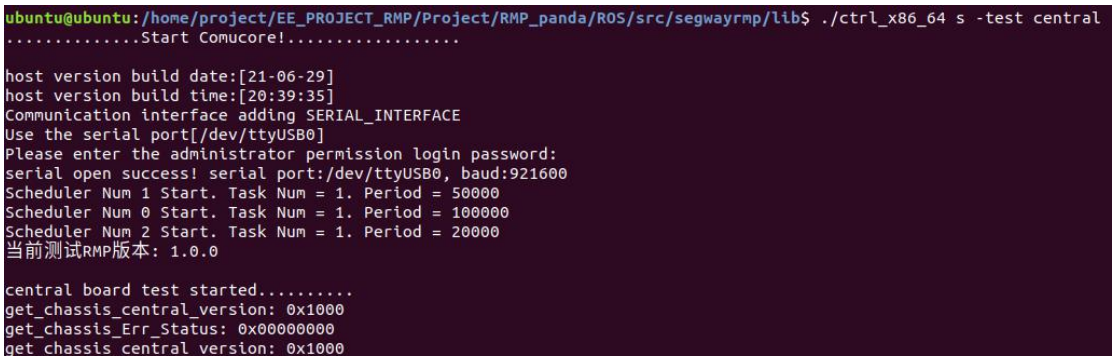


Figure 14

# Appendix 1 System Parameters and Mode switching logic

Table 1 System parameters

| | | |
|---|---|---|
| **The structural parameters** | size | Length * Width * Height (mm)<br>672 * 598 * 274 |
| | The structural parameters | Wheelbase * wheel base * ground clearance (mm) :<br>456*545*58 (chassis compressed to the lowest point) |
| | Tire size | 8" |
| | own weight | 26kg |
| | Nominal load | 28kg |
| | The disabled | 5cm/10° ramp/speed bump |
| | suspension | Independent suspension |
| | Protection grade | IPX5 |
| **The performance parameters** | Maximum speed | 3.56m/s |
| | Maximum steering speed | 2rad/s |
| | Minimum turning radius | 1.36m |
| | The braking distance | Under full load, 3.56m/s about 1m |
| | The control mode | Remote control, upper computer control |
| | The brake way | The electric brake、mechanical brake |
| **Communication** | Communication interface | UART，CAN |
| | API | ROS |
| | Feedback data | encoder, IMU |
| **The battery** | Continuous operating distance | under full load　2m/s 5、about 40km |
| | The battery | 36V 15.3Ah |
| | Charging way | Manual cable charge/quick battery change |
| **Interaction** | The keys | Emergency stop botton, power swich botton |
| | Status indication | On/off state indicator light, chassis state indicator light,<br>Control source indicator, power display, charging state display |

Table 2 Mode switching logic

| Chassis mode | input | excute | exit |
|---|---|---|---|
| Lock mode | 1. 1. The default mode for powering on the chassis<br>2. 2. Default mode after emergency stop is restored<br>3. In the vehicle control mode, it enters the mode after recoverable exceptions such as communication timeout and communication chain disconnection occur<br>4. Enter this mode after the upper computer manually forcibly clears the error | Version without brake: 0-speed closed-loop, shielding speed command, the status indicator light is always on in yellow; Version with brake: Shield the speed command, the status indicator light is always on in yellow, the brake is disabled when power off, and the brake is locked. | 1. 1. An unrecoverable exception errorcode is detected and the system enters the error mode<br>2. 2. The enable command is received and the vehicle control mode is entered<br>3. 3. The emergency stop button is triggered to enter the emergency stop mode |
| Control mode | In the lock mode, the enable command is received | Close loop, accept control instruction. Remote control car - green indicator light is often blinking; Upper control car - Indicator is green on (The version with a power-off brake will open the brake when the brake is powered on) | 1. 1. An unrecoverable exception errorCode is detected and the system enters the error mode<br>2. 2. Enter the lock mode after detecting recoverable exceptions such as communication timeout and communication chain disconnection<br>3. 3. The emergency stop button is triggered to enter the emergency stop mode |
| Emergency stop mode | In non-abnormal mode, the emergency stop button is triggered | Wheels lose power, shielding speed and | The emergency stop button is up and the lock mode is entered |

| | | | |
|---|---|---|---|
| | | enable command, status indicator is red and often flashes.(The version with the power-off brake is disabled at this time, and the brake is locked) | |
| Error mode | An unrecoverable exception errorcode was detected | Brake, wheels lose power shielding speed and enable command. Indicator light steady red (The version with the power-off brake is disabled at this time, and the brake is locked) | 1. Reboot<br>2. The upper computer manually forcibly clears the error. (manual forced clearing of errors needs to be implemented very carefully) |

# Appendix 2 Connector welding instructions

一、 The preparatory work

1、Tools

Electric iron, solder wire

2、The material

8 Pin connector, 2 Pin connector, two AWG16 cables, and eight AWG26 cables, as shown in Figure 1.

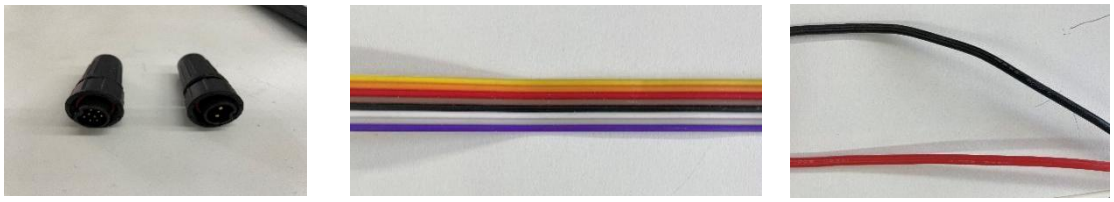

Figure 1

二、 Welding instructions (taking 8 Pin connector as an example)

1. The 8 Pin connector received by the customer is shown in Figure 2. Screw the connector from the position shown in the red arrow to the state shown in Figure 3;

Figure 2

Figure 3

2. Take out the component shown in Figure 4, which is the component 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 part. Then rotate the part 180°, which is the part to be welded;

Figure 5

4. Weld the wires of AWG26 according to the definition of PIN Angle in the welding instructions (see Appendix 3 for details). After the welding is completed, see Figure 6;

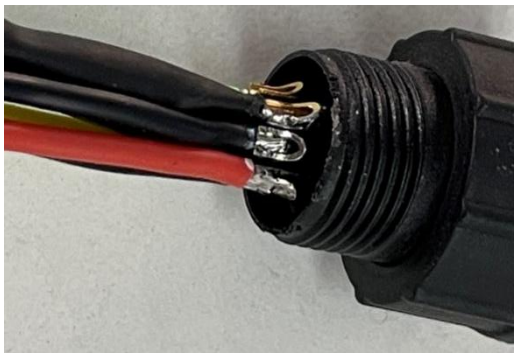

Figure 6

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

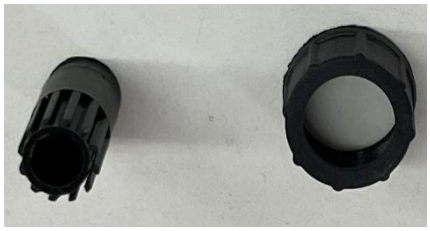

Figure 7

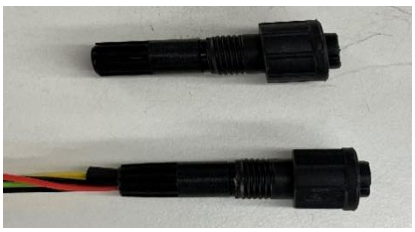

Figure8

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

Figure 9

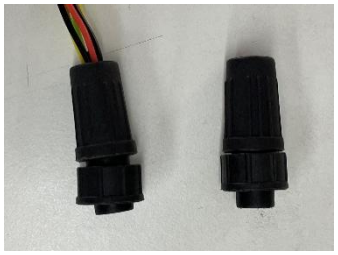

Figure 10

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

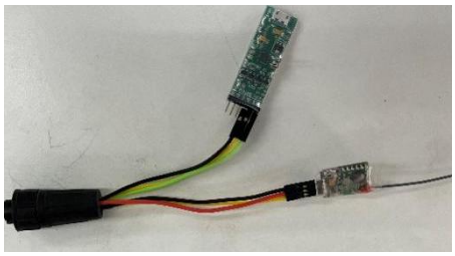

Figure 11

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

# Appendix 3 Defines the connector Pin Angle

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

# Appendix 4 C/C++ API reference documentation

- 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(s*aprctrl*datastamped*t* 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 upper computer node ID

Motor0 front_left motor node ID

Motor1 front_right motor node ID

Motor2 rear_left motor node ID

Motor3 rear_right motor node ID

Central central_board node ID

BMS batter 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 getIapProgress (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 (motor_index_e motor_index )

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

Parameter: Parameter is one of the following enumerated values:

Motor_front: Represents the front wheel circuit board

Motor_rear: Represents the rear wheel circuit board

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_calibrate_mid_value_status(void)

Function: Query if the median headway Angle has been calibrated

Parameter: None

Return value: 1: calibrated;

0: no calibrated.

- 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 - 2.3

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 -0.85 - 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 - 2

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 setHostIapCanceled (void)

Function: Interrupt the IAP upgrade process.

Parameter: none

Return value: none

- uint8_t set_calibrate_mid_value(void)

Function: Sets the command to calibrate the median Angle of the front wheel of the chassis.

Parameter : none.

Return value: 0: successfully set; other: setup failed.

- uint8_t reset_host_power_time_s(uint16_t reset_time_s);

Function: Set the reset time after power failure of the upper machine

Parameter : reset_time_s: reset interval time.

Range: 0~65535. unit: second.

Return value: 0: successfully; other: failed.