ESPrtk 3.9.8.9 Datasheet Ver 1.8.9 – BETA

Date : 19/3/2021 .
Status release : Update release.

Warning for references datasheet.

We are not responsible for any damage or incidents caused by the use of this document by users, please make sure you download it from address http://esprtk.com or http://esprtk.wap.sh or receive this document via email address contact@esprtk.com or esprtk@gmail.com of us.
In the course of using this document, if you discover any content that you believe is typographical, syntax, grammatical, or the difference of information provided in the datasheet compared to the actual behavior of hardware, etc. please report to us via email address contact@esprtk.com or esprtk@gmail.com.

Disclaimer and Copyright Notice

All contents related to the design of ESPrtk are included in this document such as diagrams, images, data transmission protocols, control command frames, ESPrtk command message frames, ESPrtk command message format , the control model, the hierarchical model, the ESPrtk data encryption algorithm .. copyright belong to the ESPrtk development team.

Information in this document, including URL references, is subject to change without notice.
THIS DOCUMENT IS PROVIDED AS IS WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.
All liability, including liability for infringement of any proprietary rights, relating to use of information in this document is disclaimed. No licenses express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein. The Wi-Fi Alliance Member logo is a trademark of the Wi-Fi Alliance. The Bluetooth logo is a registered trademark of Bluetooth SIG. All trade names, trademarks and registered trademarks mentioned in this document are property of their respective owners, and are hereby acknowledged.

Release and revision version

Rev	Date	Firmware	Detail
V1.0	4/2/2020	3.0.0	First Release
V1.2	8/3/2020	3.2.0	1. Add support Deep Sleep mode (Hardware Control mode)
V1.2	2/4/2020	3.2.5	1. WiFi Radio Range up to 2Km
V1.3 + V1.4	11/7/20202 4/7/2020	3.8.0 3.9.0	1. Change GPIO pin name. 2. Support UART_Base_to_UART_Rover mode . 3. New support GSM Cellular 2G 3G 4G LTE . 4. New support Ethernet module. 5. More support other action for Rover Repeater mode. 6. Changed Read/Write configure of ACTION_PLANNING ,INTERNET , NTRIP_CLIENT , MQTT_RTK with more support. 7. Add new Read/Write configure for INTERNET , ACTION_PLANNING ,GSM_CELLULAR , STREAM_MANAGER , POSITION_ANTENNA. 8. Hardware Control : Support W5XX for 'Control WIFI and Ethernet module connect to router'
V1.5	31/7/2020	3.9.0 +3.9.2	1. Correct circuit diagram and power voltage for Neopixel LED.
V1.6 V1.7	16/7/2020 17/7/2020	3.9.5 3.9.5	1. Changed Read/Write configure of YOUR_PROFILE (for support change CSS theme of WEB Configure interface). 2. Support view NMEA/GGA for "No Main Action" mode. 3. Hardware Control : Correct Command for Bluetooth Read String+Hex.
V1.8	16/10/2020	3.9.8	1. Changed Read/Write configure of ACTION_PLANNING with more support. 2. Hardware Control mode: Support Auto calibrate magnetometer by command . 3. Web Configure mode : Support Auto calibrate magnetometer using Sensor or Text field as input data. 4. Main Action mode: Support send IMU data to Bluetooth device .
V1.82	17/12/2020	3.9.82	1. Changed Read/Write configure of ACTION_PLANNING and POSITION_ANTENNA with more support. 2. Add new Read/Write configure for WIFI_TCP_UDP . 3. Support Send IMU_1/IMU_2 Data to WiFi TCP/UDP device . 4. Support Send RX-UART-RTK data to WiFi TCP/UDP device. 5. Support Respond realtime command control on RX-Bluetooth device . 6. Support Respond realtime command control on WiFi TCP/UDP device. 7. Stable IMU data at high speed. 8. WiFi TCP/UDP support : (TCP Server AP - TCP Server STA - TCP Client STA - UDP Server AP - UDP Server STA - UDP Client STA).

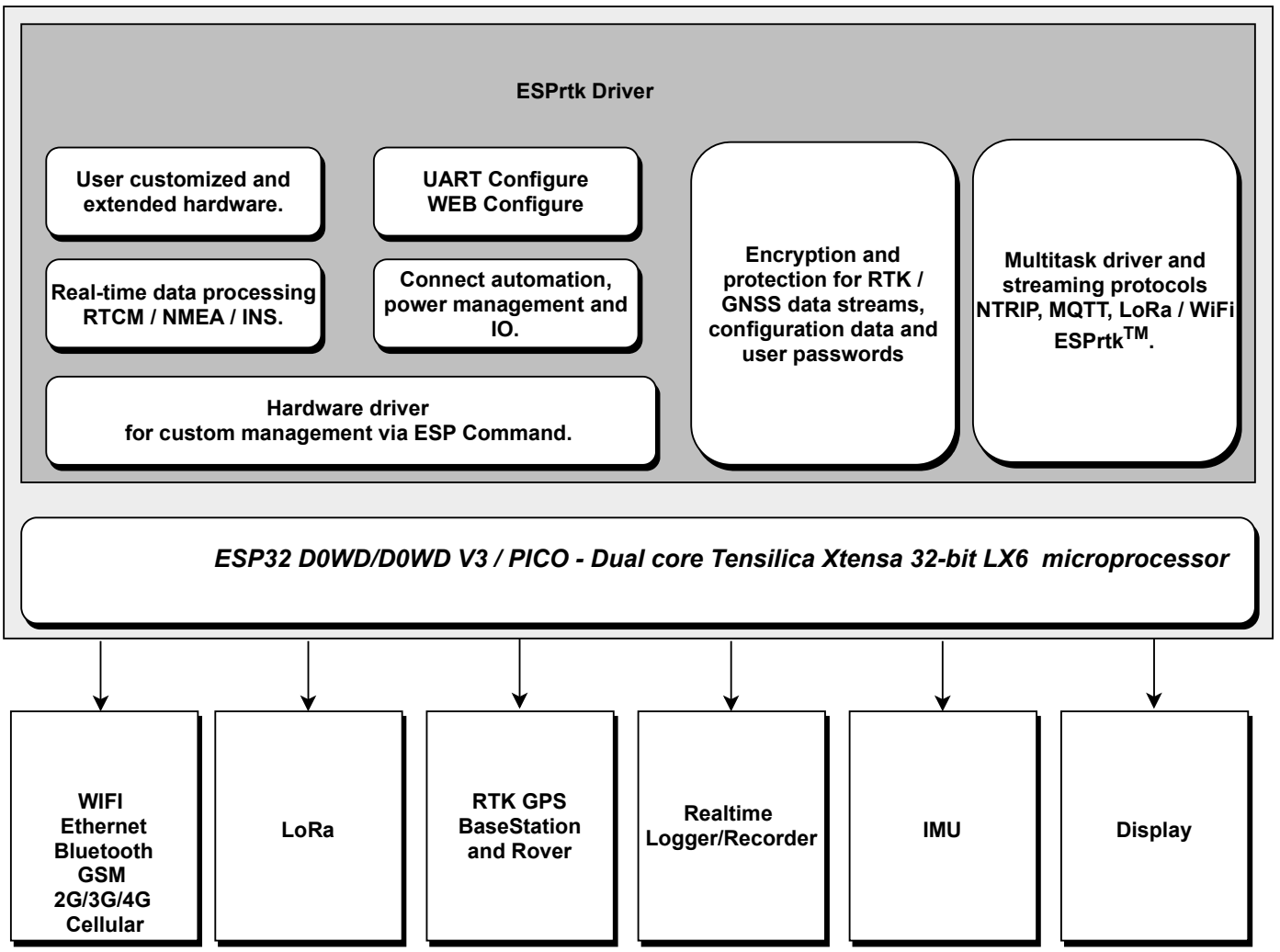
Rev	Date	Firmware	Detail
V1.87	1/2/2021	3.9.87	1. Fully Skytraq/Navspark PX1122R support Configure/Query 2. Add more support Navspark Configure/Query feature. 3. NTRIP CASTER support. 4. Changed Read/Write configure of ACTION_PLANNING. 5. Add new Read/Write configure for NTRIPCASTER_Admin and NTRIPCASTER_ACC .
V1.88	5/3/2021	3.9.88	1. Supports RTK stream insertion / injection of RTCM messages 1005,1006,1007,1008,1032,1033,1004,1012 . 2. Changed Read/Write configure of POSITION_ANTENNA.
V1.89	19/3/2021	3.9.89	1. Add new feature UART_to_WIFI_Bridge for Main Action. 2. Changed Read/Write configure of ACTION_PLANNING.

GENERAL DESCRIPTION

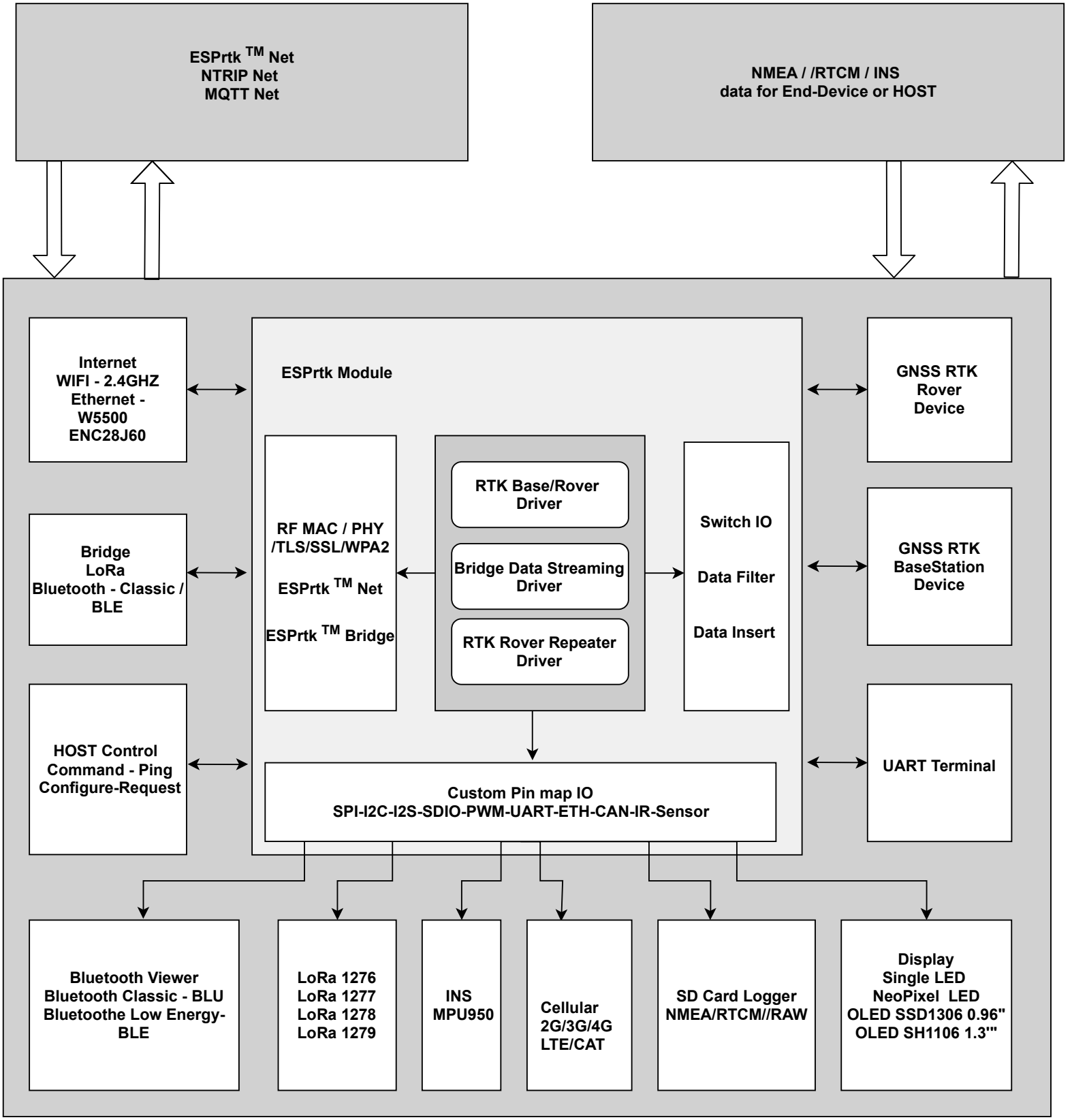
- ESPrtk module used to transmit wireless GNSS data in high precision positioning applications.
- ESPrtk is geared towards the ability to use high quality, independent, wireless and low cost platforms.
- Users can access and control ESP32 chip internal hardware.
- User update firmware.
- Multi-connection.
- Control and high speed real-time data processing.
- Protect device - Encrypt data.
- Automation of connections .
- Energy / Power / Speed / Hardware / Device manager.

KEY PRODUCT FEATURES

- MQTT™ - NTRIP CASTER™ - NTRIP™ - RADIO™ – TCP/UDP™ - bridge for RTK position.
- Basestation RTK, Rover RTK.
- RTCM data – NMEA data – UBX data- Skytraq data.
- WIFI - Bluetooth Classic - Bluetooth BLE.
- Ethernet ENC28J60 , W5500, W5xx.
- LoRa 1276/-7/-8/-9 .
- SD Card - OLED Display - IMU MPU9250 - NEOPIXEL led.
- UART Configure – ESPrtk command.
- GSM Cellular 2G/3G/4G LTE/CAT : Quectel ,SimCom, Ublox,Xbee.
- Support SIMCom 2G SIM800 + SIM900 series (SIM800A, SIM800C, SIM800L, SIM800H, SIM808, SIM868 , SIM900A, SIM900D, SIM908, SIM968) .
- Support SIMCom 3G SIMCom WCDMA/HSPA/HSPA+ Modules (SIM5360, SIM5320, SIM5300E, SIM5300E/A) SIMCom LTE Modules (SIM7100E, SIM7500E, SIM7500A, SIM7600C, SIM7600E) SIMCom SIM7000E/A/G CAT-M1/NB-IoT Module
- Support Quectel 3G Quectel BG96 Quectel M95 Quectel MC60
- Support U-blox 2G, 3G, 4G, and LTE Cat1 Cellular Modems (many modules including LEON-G100, LISA-U2xx, SARA-G3xx, SARA-U2xx, TOBY-L2xx, LARA-R2xx, MPCI-L2xx) u-blox LTE-M/NB-IoT Modems (SARA-R4xx, SARA-N4xx, but NOT SARA-N2xx)
- Digi Xbee 2G/3G/4G LTE.
- WEB Configure – UI WEB interface.
- Encrypt : Bitwise, AES , WAP, WAP2. SSL.
- Navspark : NS-HP-BD / -GL / -GN , PX1120R , PX1122R .
- Ublox : NEO , LEA, M8P, F9P, F9K,F9H.
- Hemisphere: Eclipse P307 .
- NVS Technologies:NV08C-RTK .
- Swift : Piksi Multi .


ESPrtk Driver
User customized and extended hardware.
UART Configure WEB Configure
Real-time data processing RTCM / NMEA / INS.
Connect automation, power management and IO.
Hardware driver for custom management via ESP Command.
Encryption and protection for RTK / GNSS data streams, configuration data and user passwords
Multitask driver and streaming protocols NTRIP, MQTT, LoRa / WiFi ESPrtk™.
ESP32 D0WD/D0WD V3 / PICO - Dual core Tensilica Xtensa 32-bit LX6 microprocessor
WIFI Ethernet Bluetooth GSM 2G/3G/4G Cellular
LoRa
RTK GPS BaseStation and Rover
Realtime Logger/Recorder
IMU
Display

Categories	Items	Specifications
Certification	RF Certification	FCC/CE-RED/IC/TELEC/KCC/SRRC/NCC
	Wi-Fi Certification	Wi-Fi Alliance
	Bluetooth certification	BQB
	Green Certification	REACH/RoHS
Test	Reliablity	HTOL/HTSL/uHAST/TCT/ESD
Wi-Fi	Protocols	802.11 b/g/n (802.11n up to 150 Mbps)
		A-MPDU and A-MSDU aggregation and 0.4 µs guard interval support
Bluetooth	Frequency range	2.4 GHz ~ 2.5 GHz
	Protocols	Bluetooth v4.2 BR/EDR and BLE specification
	Radio	NZIF receiver with –97 dBm sensitivity
		Class-1, class-2 and class-3 transmitter
		AFH
	Audio	CVSD and SBC
Hardware	Module interfaces	SD card, UART, SPI, SDIO, I2C, LED PWM, Motor PWM, I2S, IR, pulse counter, GPIO, capacitive touch sensor, ADC, DAC
	On-chip sensor	Hall sensor
	Integrated crysta	40 MHz crysta
	Integrated SPI flash 1	4 MB
	Operating voltage/Power supply	3.0 V ~ 3.6 V
	Operating current	Average: 80 mA
	Minimum current delivered by power supply	500 mA
	Recommended operating temperature range	–40 °C ~ +85 °C
	Moisture sensitivity level (MSL)	Level 3


ESPrtk TM Net
NTRIP Net
MQTT Net
NMEA / /RTCM / INS
data for End-Device or HOST
Internet
WIFI - 2.4GHZ
Ethernet -
W5500
ENC28J60
ESPrtk Module
GNSS RTK
Rover
Device
Bridge
LoRa
Bluetooth - Classic /
BLE
RF MAC / PHY
/TLS/SSL/WPA2
ESPrtk TM Net
ESPrtk TM Bridge
RTK Base/Rover
Driver
Bridge Data Streaming
Driver
RTK Rover Repeater
Driver
Switch IO
Data Filter
Data Insert
GNSS RTK
BaseStation
Device
HOST Control
Command - Ping
Configure-Request
Custom Pin map IO
SPI-I2C-I2S-SDIO-PWM-UART-ETH-CAN-IR-Sensor
UART Terminal
Bluetooth Viewer
Bluetooth Classic - BLU
Bluetoothe Low Energy-
BLE
LoRa 1276
LoRa 1277
LoRa 1278
LoRa 1279
INS
MPU950
Cellular
2G/3G/4G
LTE/CAT
SD Card Logger
NMEA/RTCM//RAW
Display
Single LED
NeoPixel LED
OLED SSD1306 0.96"
OLED SH1106 1.3'"

Overview Introduction

Table of Contents

Overview Introduction

About ESPrtk

ESPrtk is an abbreviation for ESP32 RTK.

ESPrtk is a small circuit board used to transmit wireless GNSS data in high precision positioning applications.

ESPrtk is geared towards the ability to use high quality, independent, wireless and low cost platforms.

ESPrtk's heart is a powerful ESP32 processor .

ESPrtk is a data bridge, it provides many solutions to deploy network models focusing on GNSS data transmission (such as RAW / RTCM / NMEA) of RTK location applications. Along with that is the maximum exploitation of the hardware power of ESP32 to provide the most powerful and intelligent features..

Control System

ESPrtk is designed to run Hierarchical Control System (HCS).

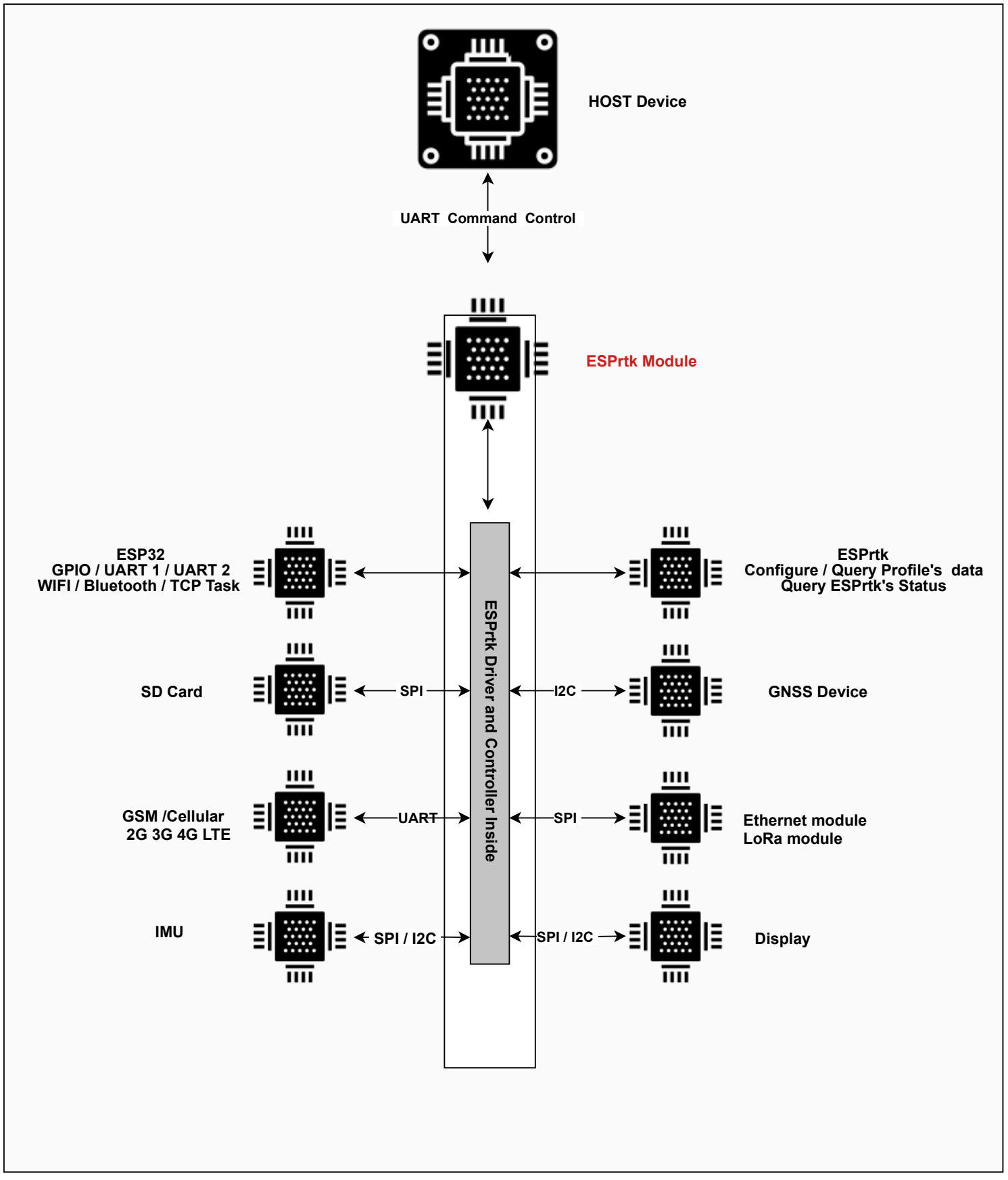
In particular, ESPrtk is a data bridge for GNSS / RTK message transmission, which integrates many programs running in parallel (Driver) to connect to many peripheral devices such as GNSS Receiver, IMU, SD Card, Display, Ethernet ...etc . ESPrtk now is an external controller.

Finally, ESPrtk is controlled, managed and monitored by a higher level control device called HOST . HOST can be a PC or maybe a microcontroller . ESPrtk is now the periphery of HOST.

A special feature of the ESPrtk design is that HOST not only controls ESPrtk, but HOST can also manage and control all peripherals devices connected to ESPrtk such as GNSS Receiver, IMU, SD Card, Display, Ethernet. ..

With this design, ESPrtk will reduce a huge amount of programs running on HOST, making the deployment of RTK systems easier.

The connection between HOST and ESPrtk is done on ESPrtk's UART_0 port. HOST communicates, configures and controls with ESPrtk (or peripheral) through the exclusive ESPrtk Command (ESP Command).

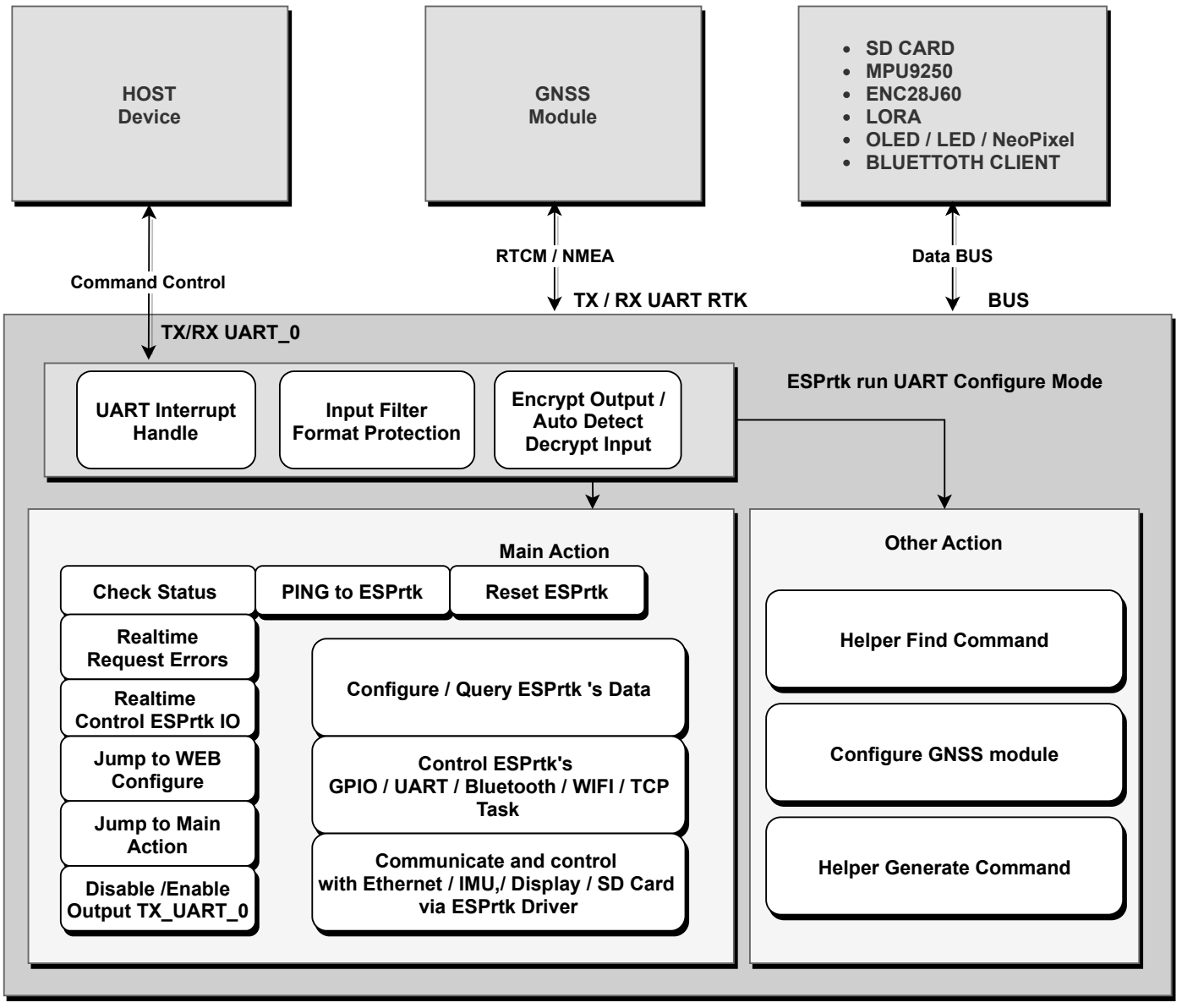

HOST Device
UART Command Control
ESPrtk Module
ESP32
GPIO / UART 1 / UART 2
WIFI / Bluetooth / TCP Task
ESPrtk
Configure / Query Profile's data
Query ESPrtk's Status
SD Card
SPI
I2C
GNSS Device
GSM /Cellular
2G 3G 4G LTE
UART
SPI
Ethernet module
LoRa module
IMU
SPI / I2C
SPI / I2C
Display
ESPrtk Driver and Controller Inside

Actions Mode

ESPrtk Designed around applications related to positioning and collecting RTK data

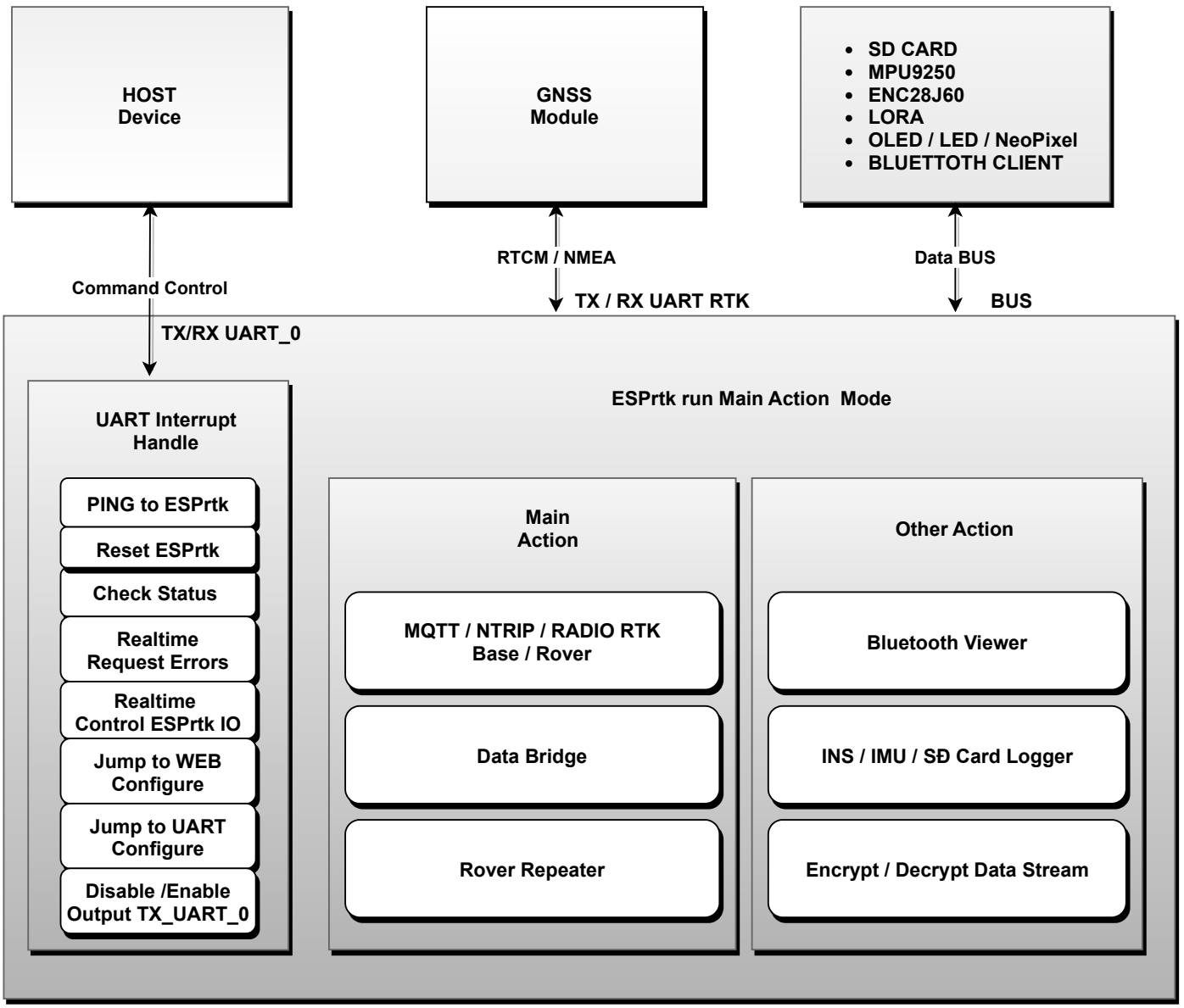
With 3 different independent operating modes,

1. **UART Configure Mode** : In this mode, HOST communicates with ESPrtk via UART_0 port to configure (change settings) on ESPrtk. HOST can also control ESPrtk's hardware features such as controlling IO pins, Bluetooth, WIFI, TCP Task, etc., or controlling other peripheral devices connected to ESPrtk.

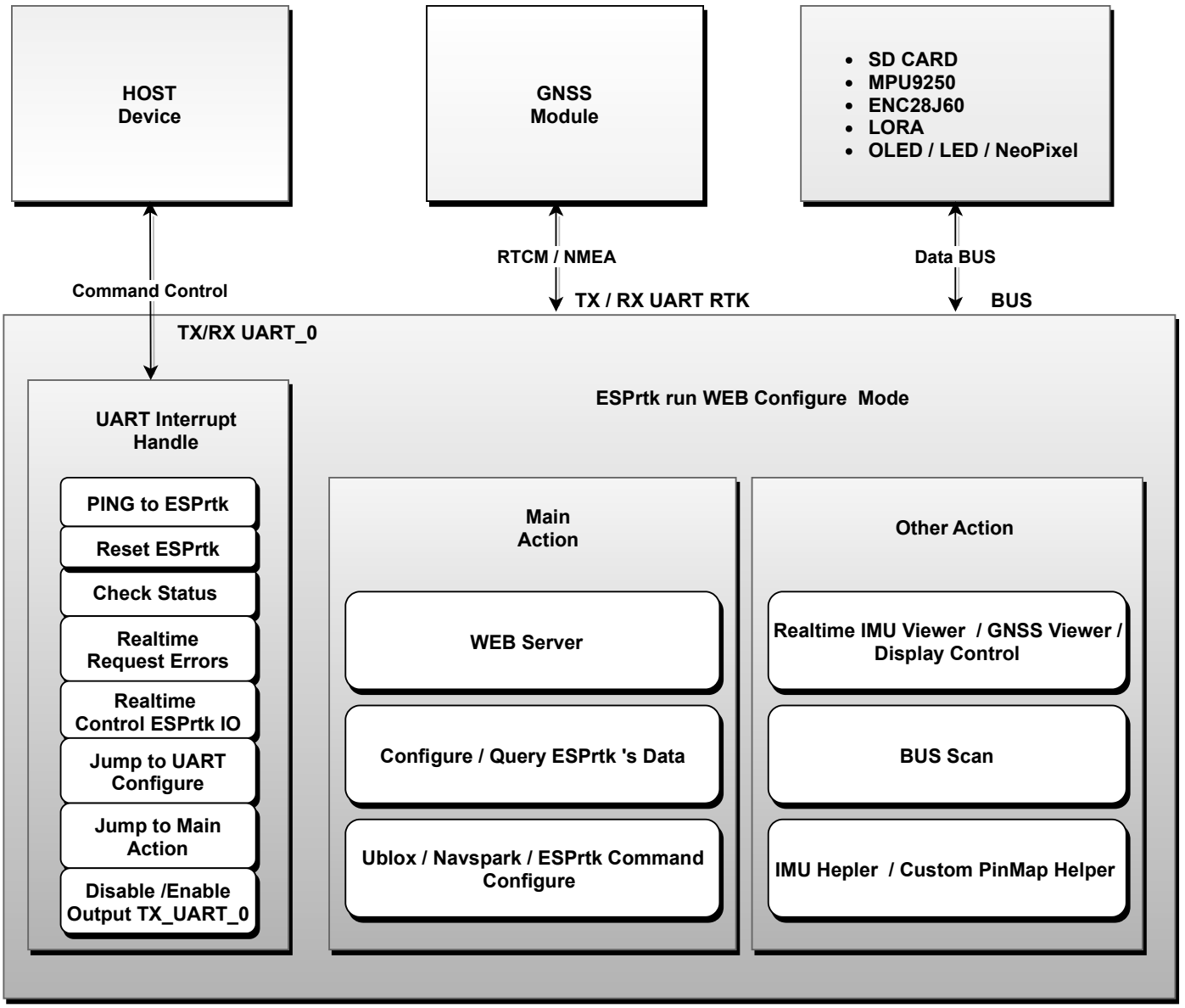


2. **Main Action Mode**: After configuring ESPrtk, the changed data will be stored in ESPrtk's FLASH memory and they will not be lost during power outages. Once in Main Action mode, ESPrtk reads the boot instructions and performs the main activities planned in advance by the user (via UART Configure or WEB Configure).

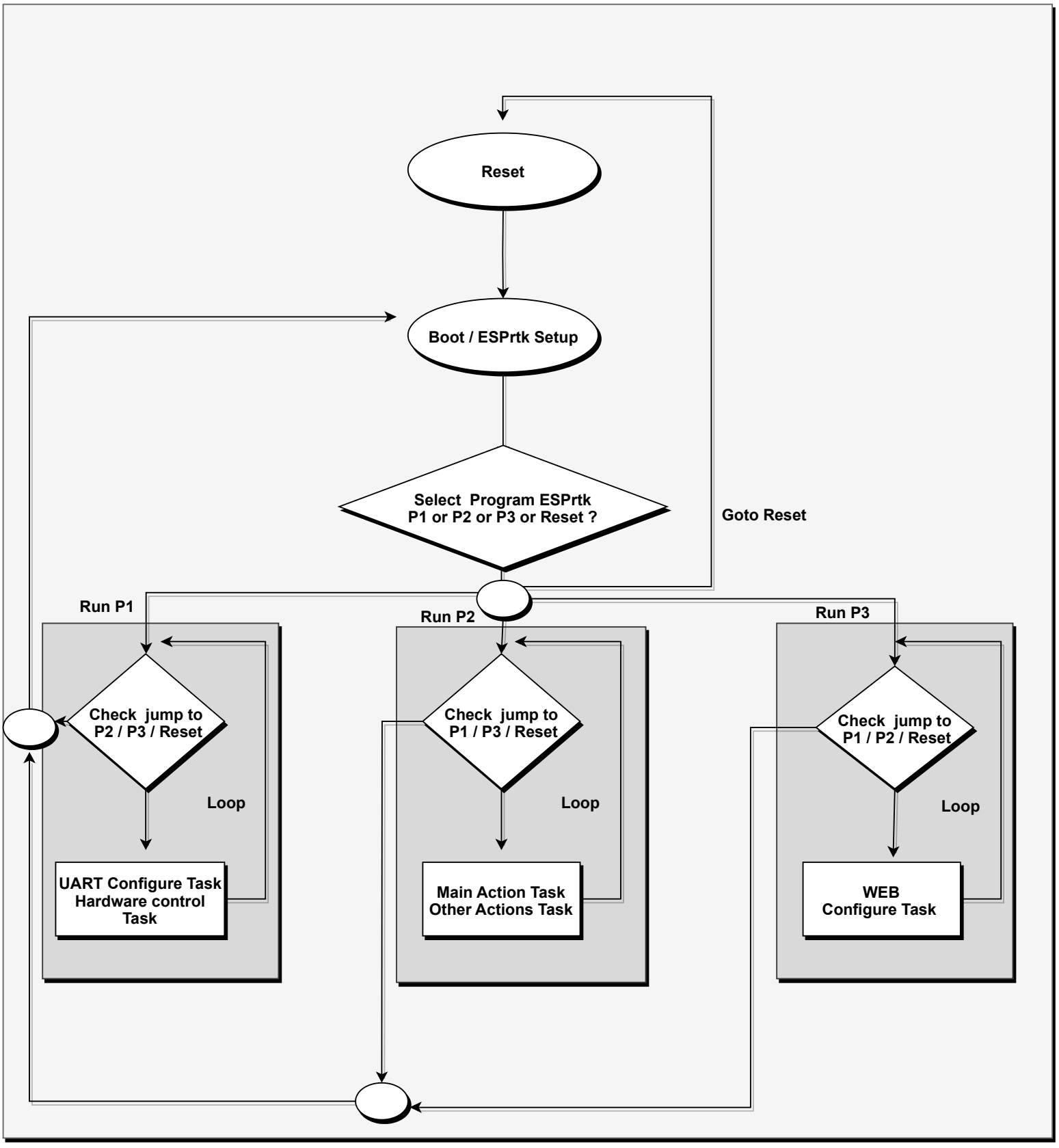
   Main Action is central mode because most applications that use ESPrtk actually use it.



3. **WEB Configure Mode**: In addition to configuring ESPrtk via UART, users can perform ESPrtk configuration via an intuitive WEB UI interface using any WEB browser with a web server run by ESPrtk over a WIFI connection. Along with that are tools that help users connect to peripherals such as GNSS Reveiver, IMU, SD, Display ..



The above three operating modes operate independently of each other and HOST can switch between them with a control command at any time while ESPrtk running the program.

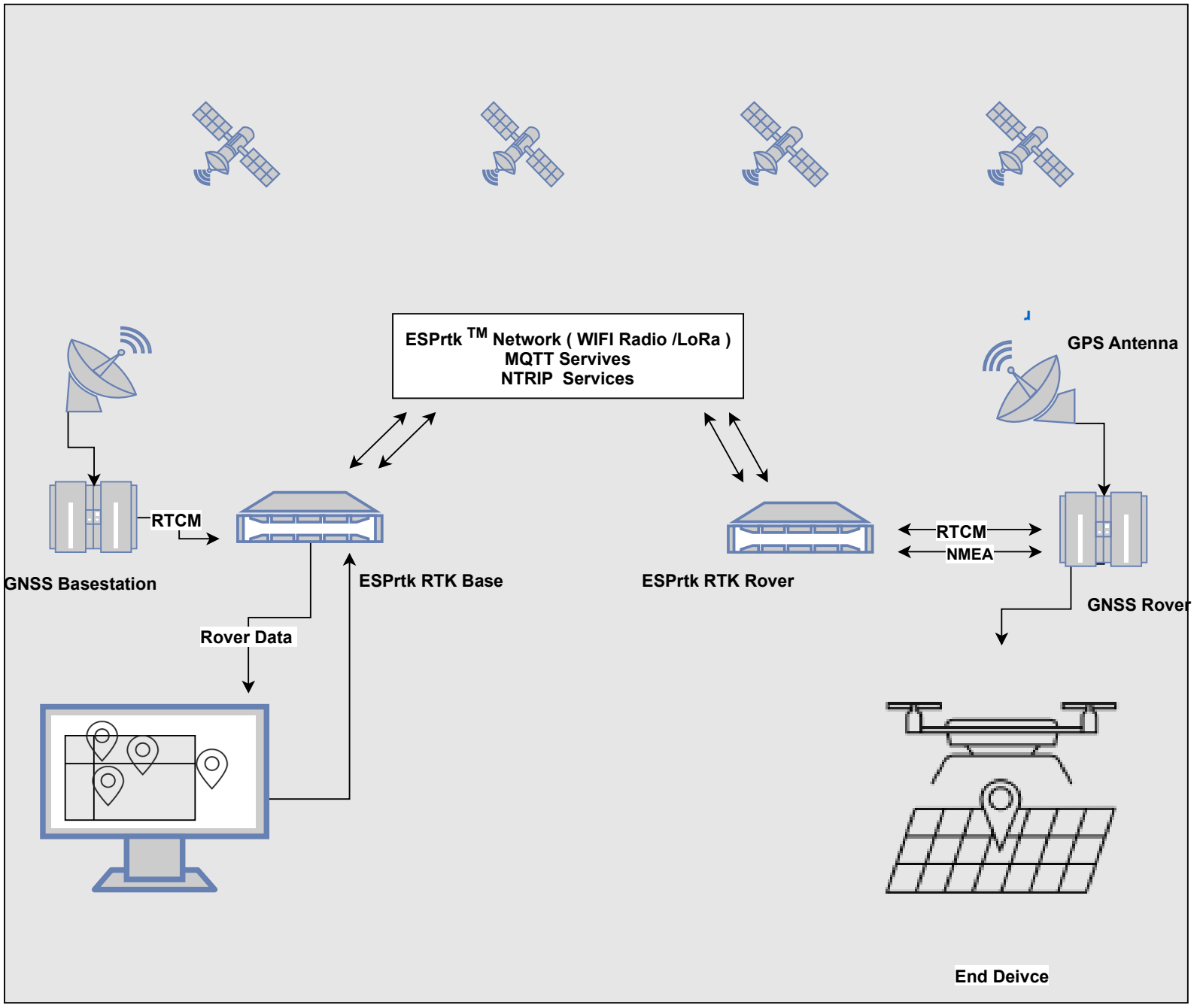


ESPrtk Run in Main Action Mode

ESPrtk with RTK Position Applications

In RTK positioning, GNSS (Rover) receivers need RAW / RTCM data from other GNSS receivers (BaseStation). In most practical applications, Rover is always on the move and the distance from Rover to BaseStation is large (in kilometers).

ESPrtk is used as a data link bridge between BaseStation and Rover.



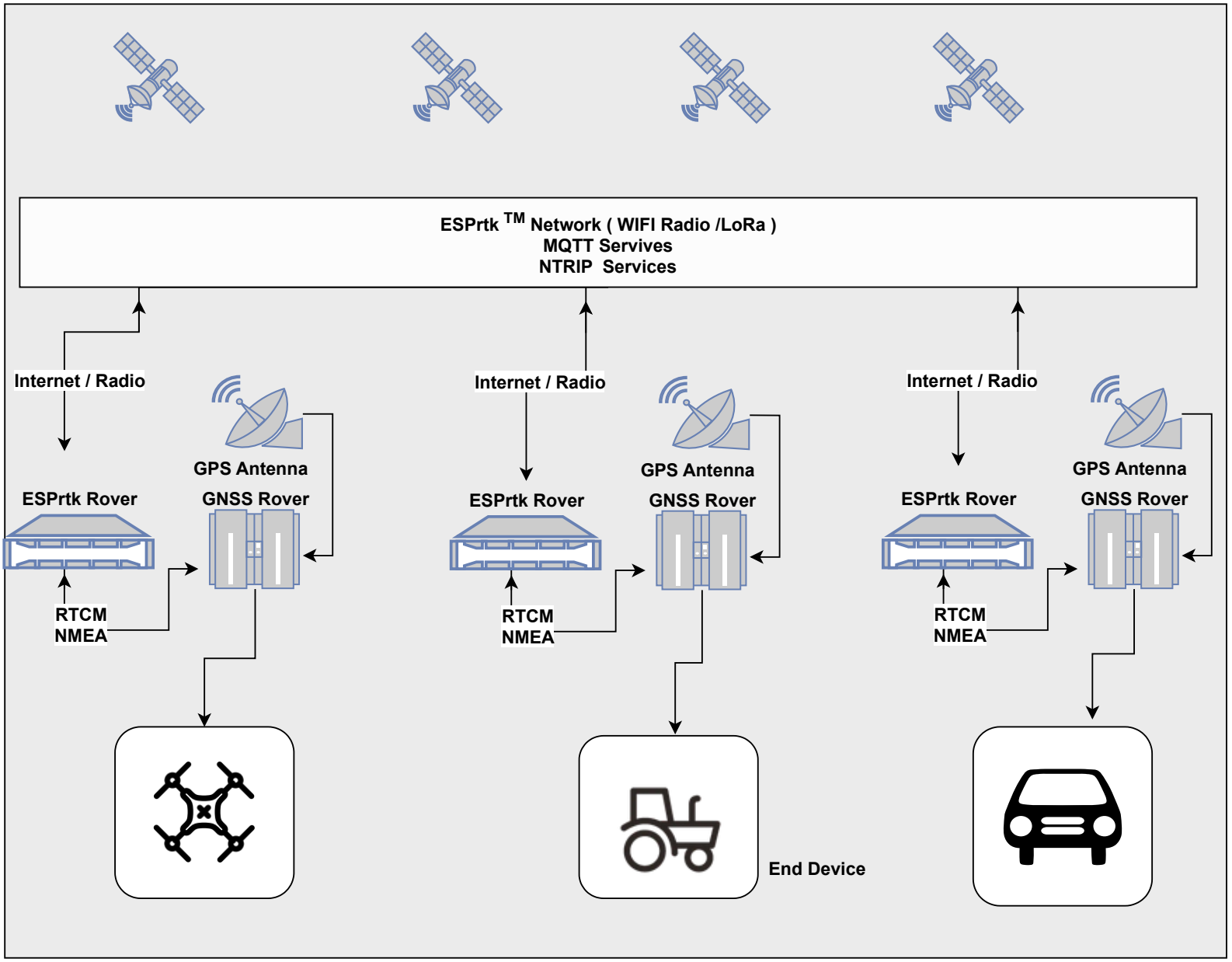
If set to a Base, ESPrtk will collect RTCM / RAW data from one of the UART ports and then send it over the wireless interface to one or more other devices (Rover).

If set to Rover, the data received from the Base will be streamed to the UART port to the RTK module.

There may be more than one Rover using RTCM / RAW data from Base at the same time.

At the same time, ESPrtk Rover can send back NMEA data (such as $ GPGGA) containing positioning information for ESPrtk Base and will be exported on TX_UART0 port. HOST devices can manage Rover's location through this feature.

When combined with ESPrtk, the cost of initial investment and system operation will be greatly saved.



ESPrtk applies both types of link: Internet and Radio.

With the Internet, ESPrtk supports two popular types of RTCM transmission protocols, MQTT and NTRIP via WiFi or Ethernet connection. With this type of link, the problem of distance has been solved.

With Radio, ESPrtk uses its own 2.4Ghz WIFI signal (for distances less than 3km) or frequency 443Mhz -> 998Mhz when combining ESPrtk with LoRa module (for distances less than 50km).

Configure ESPrtk run for RTK Position Application

The full RTK model includes 1 ESPrtk Base and 1 (or more) ESPrtk Rover linked together by MQTT / NTRIP (online) or via Radio link (offline).

When configuring ESPrtk, they will be configured in pairs.

- **MQTT RTK Base** and **MQTT RTK Rover (or MQTT Rover Repeater)** for MQTT link.
- **NTRIP RTK Base** and **NTRIP RTK Rover (or NTRIP Rover Repeater)** for NTRIP link.
- **NTRIP CASTER Server** for NTRIP link.
- **Radio RTK Base** and **Radio RTK Rover** for Radio link.

On version 3.9.8.7, the features are supported as shown in the table below.

Main Action	Detail
MQTT RTK Base	**Send data (from RX-UART-RTK) to (MQTT Broker)**
MQTT RTK Rover	**Export data (from MQTT Broker) to TX-UART-RTK port**
NTRIP RTK Base	**Send data (from RX-UART-RTK) to (NTRIP Caster)**
NTRIP RTK Rover	**Export data (from NTRIP Caster) to TX-UART-RTK port**
NTRIP CASTER Server	**Get RTCM/RAW /.. data from RX-UART-RTK (RX1/RX2) then send to NTRIP Client (Rover).**
	Get RTCM/RAW /.. data from NTRIP Server (BaseStation) then send to NTRIP Client (Rover).
Radio RTK Base	**Send data (from RX-UART-RTK) to (RTK Radio_Rover)**
Radio RTK Rover	**Export data (from RTK Radio_Base) to TX-UART-RTK port**

Other Action for RTK Base	MQTT RTK Base	NTRIP RTK Base	Radio RTK Base
Export IMU data to TX_UART_0 port	●	●	●
Send RTCM/RAW/.. data (from RX-UART-RTK) to Bluetooth Client	●	●	●
Save RTCM/RAW/.. data (from RX-UART-RTK) to SD Card	●	●	●
Base-Export NMEA data (from Rovers) to TX_UART_0 port	●	◌	●
Base-Priority export NMEA Data from Rovers	●	◌	●
Send IMU_1 Data to Bluetooth Client	●	●	●
Send IMU_2 Data to Bluetooth Client	●	●	●
Send Data (from RX-UART-RTK) to WiFi TCP/UDP device	●	●	●
Send IMU_1 Data to WiFi TCP/UDP device	●	●	●
Send IMU_2 Data to WiFi TCP/UDP device	●	●	●
Respond realtime command control on Bluetooth Client	●	●	●
Respond realtime command control on WiFi TCP/UDP	●	●	●

Other Action for RTK Rover	MQTT RTK Rover	NTRIP RTK Rover	Radio RTK Rover
Export IMU Data to TX_UART_0 port	●	●	●
Save NMEA/RTCM Data (from RX-UART-RTK) to SD Card	●	●	●
Send NMEA/RTCM data (from RX-UART-RTK) to Bluetooth Client	●	●	●
Save RTCM/RAW... data (from Server/RTK_Base) to SD Card	●	●	●

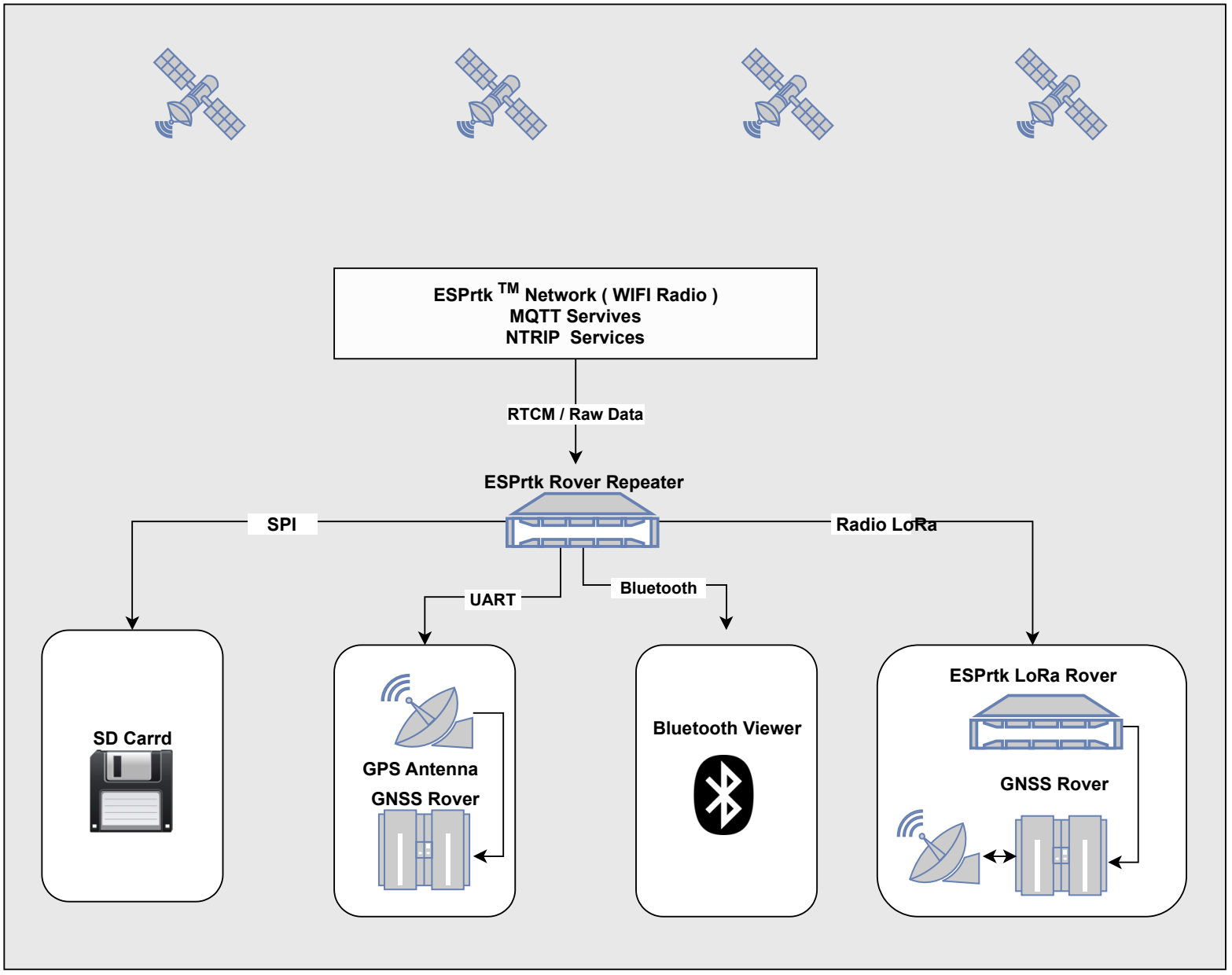
Send GGA message (from RX-UART-RTK) to Server/Radio RTK Base + Export GGA to TX_UART0	●	●	●
Send all NMEA messages (from RX-UART-RTK) to Server / Radio RTK Base + Export NMEA to TX_UART0	●	●	●
Send IMU_1 Data to Bluetooth Client	●	●	●
Send IMU_2 Data to Bluetooth Client	●	●	●
Send Data (from RX-UART-RTK) to WiFi TCP/UDP device	●	●	●
Send IMU_1 Data to WiFi TCP/UDP device	●	●	●
Send IMU_2 Data to WiFi TCP/UDP device	●	●	●
Respond realtime command control on Bluetooth Client	●	●	●
Respond realtime command control on WiFi TCP/UDP	●	●	●

Configure ESPrtk run as Rover Repeater for RTK Application

The RTK Rover Repeater has the same main function as the RTK Rover is to receive RAW / RTCM data from RTK Base and export to TX_UART_RTK port.

The RTK Rover Repeater adds a feature that repeats data from the Base. It is then a Rover (receiving data from Base) and a Base (sending data received to other Rover or to Bluetooth Clinet devices).

In ESPrtk 3.0.0 version, the repetition feature is applied to Rover when running as MQTT / NTRIP Rover, RAW / RTCM data received will be transmitted repeatedly to ESPrtk running as Radio RTK Rover.



The repetition feature is primarily exploited in applications that require the expansion of RTK networks in areas where Internet use is limited in distance.

The RTK model includes :

- 1 RTK Rover Repeater : Send RTCM/RAW (from RTK MQTT/NTRIP Base) to RTK Radio Rover .
- 1 (or more) RTK Radio Rover : Export RTCM/RAW data to GNSS Rover Receiver.

When configuring ESPrtk, they will be configured in pairs.

- **MQTT RTK Base** and **MQTT RTK Rover Repeater** for MQTT link.
- **NTRIP RTK Base** and **NTRIP RTK Rover Repeater** for NTRIP link.

And

- **MQTT RTK Rover Repeater** and **Radio RTK Rover** for Radio link.
- **NTRIP RTK Rover Repeater** and **Radio RTK Rover** for Radio link.

(When ESPrtk configure as **MQTT RTK Rover Repeater or NTRIP RTK Rover Repeater** that mean it will run as an **RTK MQTT / NTRIP Rover** and also run as an **RTK Radio Base** at the same time on one ESPrtk device)

On version 3.0.0, the features are supported as shown in the table below.

Main Action	Detail
MQTT RTK Base	**Send data (from RX-UART-RTK) to (MQTT Broker)**
MQTT RTK Rover	**Export data (from MQTT Broker) to TX-UART-RTK port**
MQTT RTK Rover Repeater	**Export data (from MQTT Broker) to TX-UART-RTK port**
NTRIP RTK Base	**Send data (from RX-UART-RTK) to (NTRIP Caster)**
NTRIP RTK Rover	**Export data (from NTRIP Caster) to TX-UART-RTK port**
NTRIP RTK Rover Repeater	**Export data (from NTRIP Caster) to TX-UART-RTK port**
Radio RTK Rover	**Export data (from RTK Radio_Base) to TX-UART-RTK port**

Other Action for RTK Rover Repeater	MQTT RTK Rover Reppeat	NTRIP RTK Rover Reppeat	Radio RTK Rover Reppeat
Export IMU Data to TX_UART_0 port	●	●	●
Save NMEA / RTCM Data (from RX-UART-RTK) to SD Card	●	●	●
Rover-Save RTCM/RAW... data (from Server/RTK_Base) to SD Card	●	●	●
Send RTCM/RAW... data (from Server/RTK_Base) to Bluetooth Client	●	●	●
Send RTCM/RAW... data (from Server/RTK_Base) to LoRa_Radio_Rover	●	●	●
Base-Export NMEA/GGA data (from Rovers) to TX_UART_0 port	●	●	●
Base-Priority export NMEA Data from Rovers	●	●	●
Rover-Send GGA message (from RX-UART-RTK) to Server/RTK_Base + Export GGA to TX_UART0	●	●	●
Rover-Send all NMEA messages (from RX-UART-RTK) to Server/ RTK_Base + Export NMEA to TX_UART0	●	●	●
Send IMU_1 Data to Bluetooth Client	●	●	●
Send IMU_2 Data to Bluetooth Client	●	●	●
Send Data (from RX-UART-RTK) to WiFi TCP/UDP device	●	●	●
Send IMU_1 Data to WiFi TCP/UDP device	●	●	●
Send IMU_2 Data to WiFi TCP/UDP device	●	●	●
Respond realtime command control on Bluetooth Client	●	●	●
Respond realtime command control on WiFi TCP/UDP	●	●	●

Configure ESPrtk - UART_Base_to_UART_Rover mode

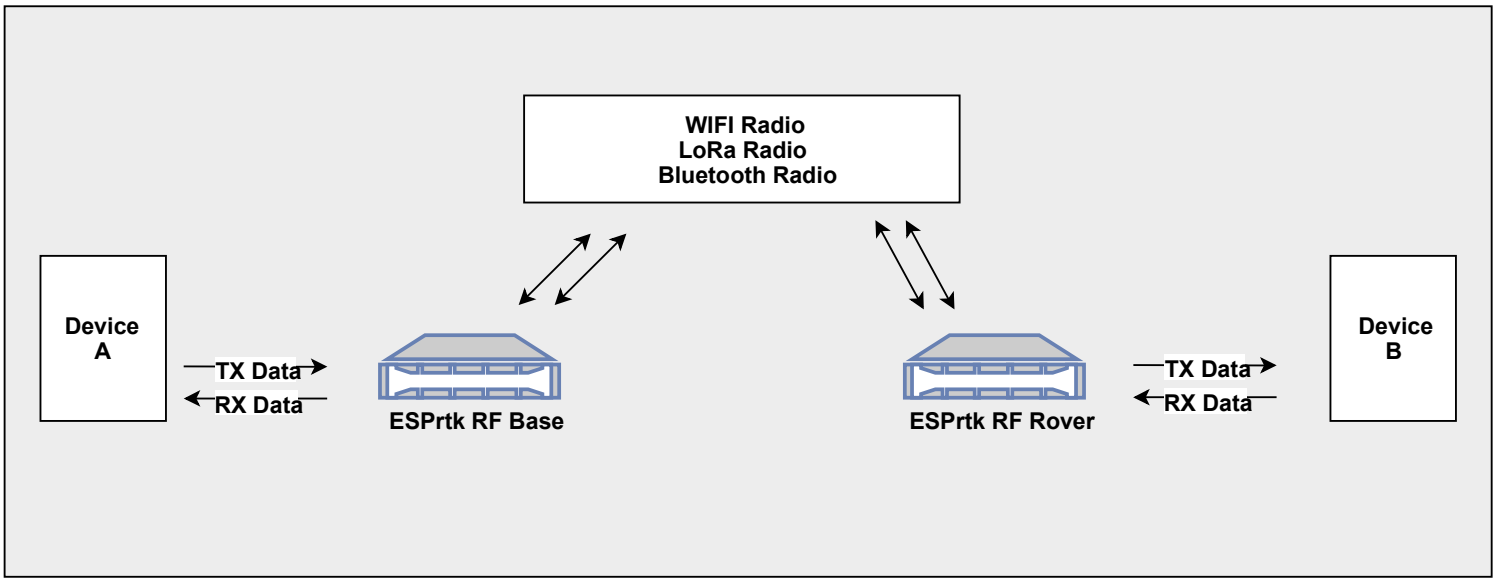
Main Action	Detail
UART_Base_to_UART_Rover	Get data (from RX-UART-RTK) as RTK Base and Export to TX-UART-RTK port as RTK Rover

Other Action for RTK Rover Repeater	UART_Base_to_UART_Rover
Export IMU Data to TX_UART_0 port	●
Rover- Save RTCM/RAW... data (from Server/RTK_Base) to SD Card (Save output TX-UART-RTK)	●
Send RTCM/RAW... data (from Server/RTK_Base) to Bluetooth Client	●
Send RTCM/RAW... data (from Server/RTK_Base) to Bluetooth Client	●
Send IMU_1 Data to Bluetooth Client	●
Send IMU_2 Data to Bluetooth Client	●
Send Data (from RX-UART-RTK) to WiFi TCP/UDP device	●
Send IMU_1 Data to WiFi TCP/UDP device	●
Send IMU_2 Data to WiFi TCP/UDP device	●
Respond realtime command control on Bluetooth Client	●
Respond realtime command control on WiFi TCP/UDP	●

Configure ESPrtk run for Data Bridge

When using ESPrtk as RTK Base / Rover it will use Broadcast link (1 Base and Rover unlimited) and optimized for RTK applications.

When using ESPrtk as Data Bridge it will use Point To Point link (1 Base and 1 Rover with peer connection) optimized for 2-way data transmission between Base and Rover similar to UART Link Radio modules. .



RTK Base / Rover and Data Bridge Base / Rover modes use 2 different link models and they will not work when combining them.

Data Bridge Base / Rover only supports connection via Radio Link or Bluetooth.

'Bluetooth to Radio' mode is compatible with 'UART to Radio'.

When configuring ESPrtk, they will be configured in pairs.

- **UART_to_RF_Base** and **UART_to_RF_Rover** .
- **Bluetooth_to_RF_Base** and **Bluetooth_to_RF_Rover**.
- **Bluetooth_to_RF_Base** and **UART_to_RF_Rover** .
- **UART_to_RF_Base** and **Bluetooth_to_RF_Rover**.

Main Action	Detail
UART_to_Bluetooth	Send data (from RX-UART-RTK) to (Blueooth-Client) + Export data (from Blueooth-Client) to TX-UART-RTK port
UART_to_RF_Base	Send data (from RX-UART-RTK) to (**UART_to_Radio_Rover**) + Export data (from **UART_to_Radio_Rover**) to TX-UART-RTK port
UART_to_RF_Rover	Send data (from RX-UART-RTK) to (**UART_to_Radio_Base**) + Export data (from **UART_to_Radio_Base**) to TX-UART-RTK port
Bluetooth_to_RF_Base	Send data (from Blueooth-Client) to (**UART_to_Radio_Rover**) + Send data (from **UART_to_Radio_Rover**) to (Blueooth-Client)
Bluetooth_to_RF_Rover	Send data (from Blueooth-Client) to (**UART_to_Radio_Base**) + Send data (from **UART_to_Radio_Base**) to (Blueooth-Client)

- When using Bluetooth, the connection between the End Devices and ESPrtk is Bluetooth, the connection between ESPrtk Base and ESPrtk Rover will be Radio WIFI (or Radio LoRa).

- When using UART , the connection between the End Devices and ESPrtk is UART, the connection between ESPrtk Base and ESPrtk Rover will be Radio WIFI (or Radio LoRa).

Main Action	Communicate type	Wireless Link
UART_to_Bluetooth	**UART**	**Bluetooth**
UART_to_RF_Base	**UART**	**Radio (WiFI / LoRa)**
UART_to_RF_Rover	**UART**	**Radio (WiFI / LoRa)**
Bluetooth_to_RF_Base	**Bluetooth**	**Radio (WiFI / LoRa)**
Bluetooth_to_RF_Rover	**Bluetooth**	**Radio (WiFI / LoRa)**

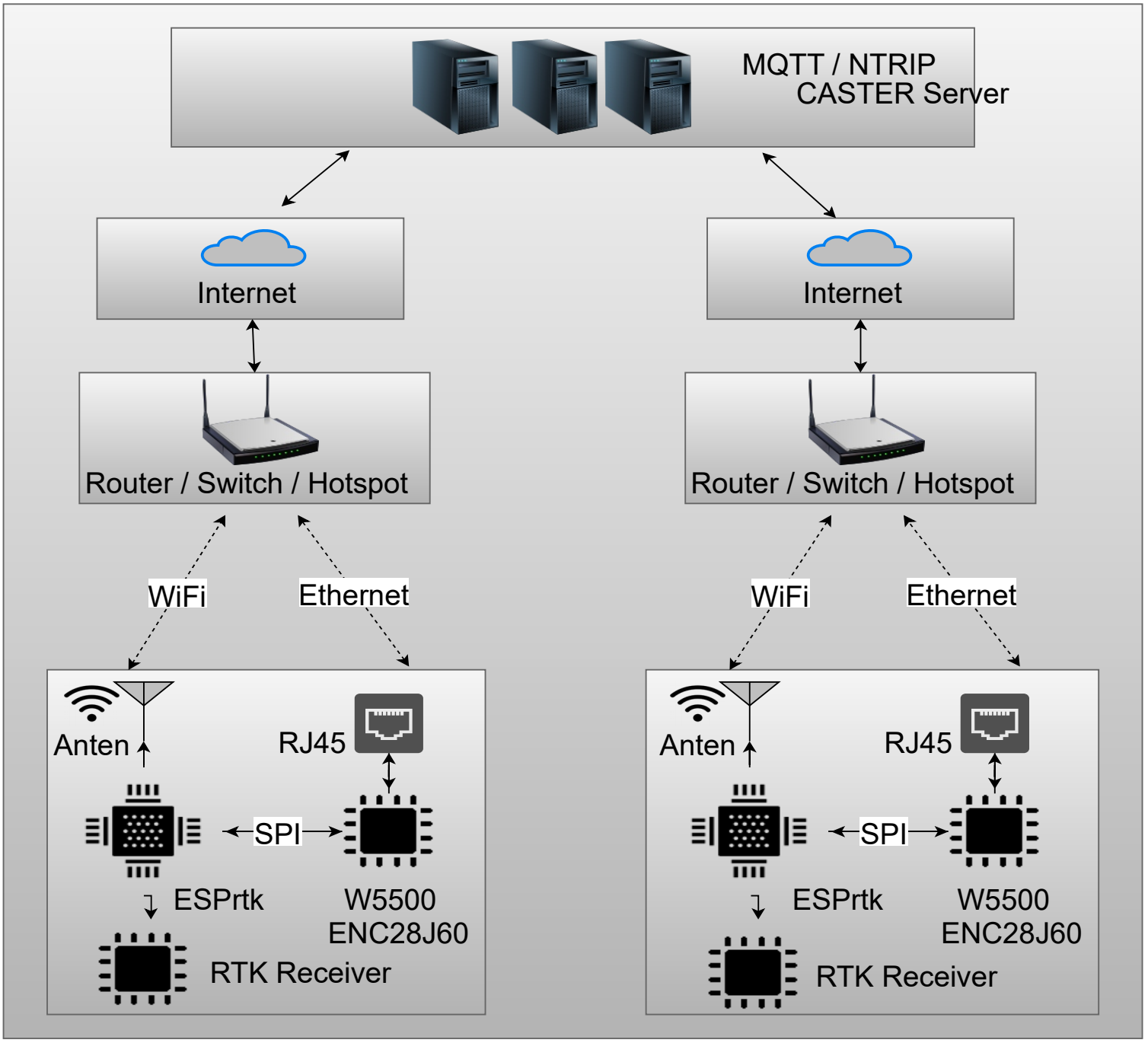
Other Action Data Bridge	Save data send (TX-Wireless-Data) to SD Card	Save data receive (RX-Wireless-Data) to SD Card
UART_to_Bluetooth	●	●
UART_to_RF_Base	●	●
UART_to_RF_Rover	●	●
Bluetooth_to_RF_Base	●	●
Bluetooth_to_RF_Rover	●	●

Configure to access Internet for MQTT/NTRIP

ESPrtk supports Internet access with both connections:

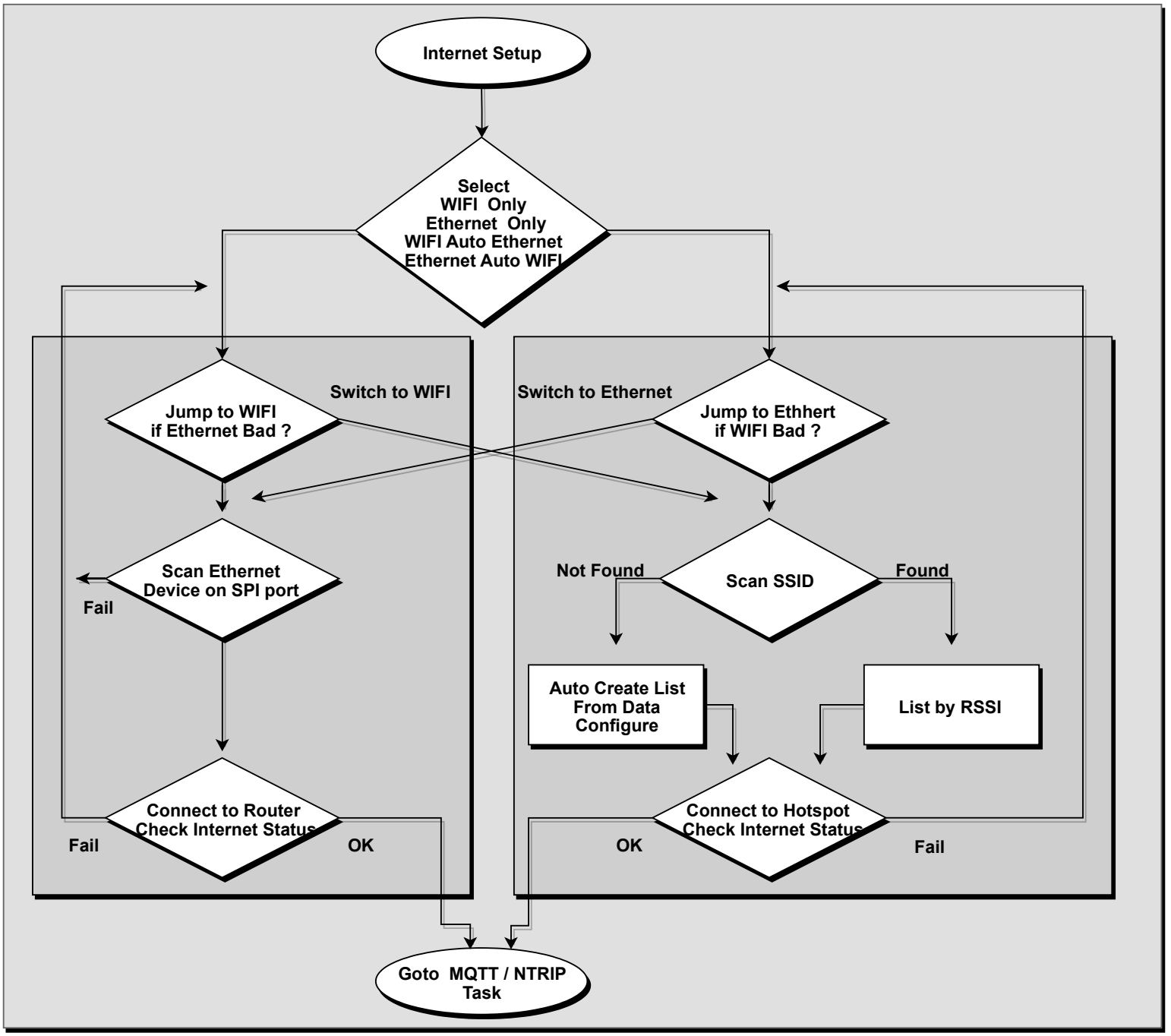
WIFI : High-speed wireless WIFI connection on ESP32 helps mobile connections for flexible Rover devices.

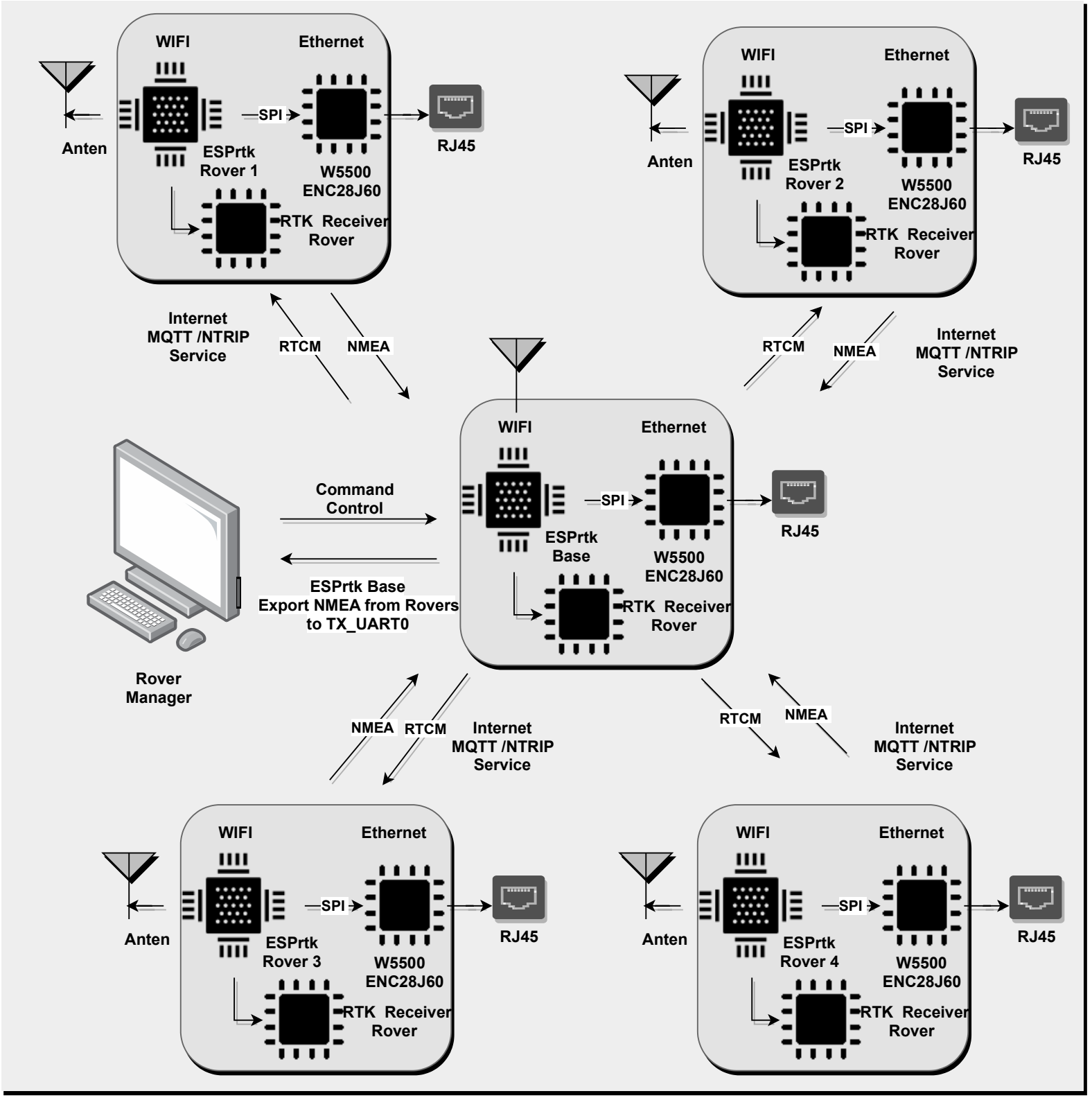
Ethernet: Internet connection will be suitable for Basestation with absolute stability. Ethernet module support are W5500 (100Mbps) , W5110 .., and ENC28J60 (10Mbps) communicate to ESPrtk via SPI port to access internet.



There are 4 operating modes:

1. WiFi : WiFi only.
2. Ethernet : Ethernet only .
3. Auto WiFi: Select WIFI for the first connect, automatically switching to Ethernet (or back to WIFI) if Internet access is not available after 60 seconds.
4. Auto Ethernet: Select Ethernet for the first connect, automatically switching to WIFI (or back to Ethernet) if Internet access is not available after 60 seconds.

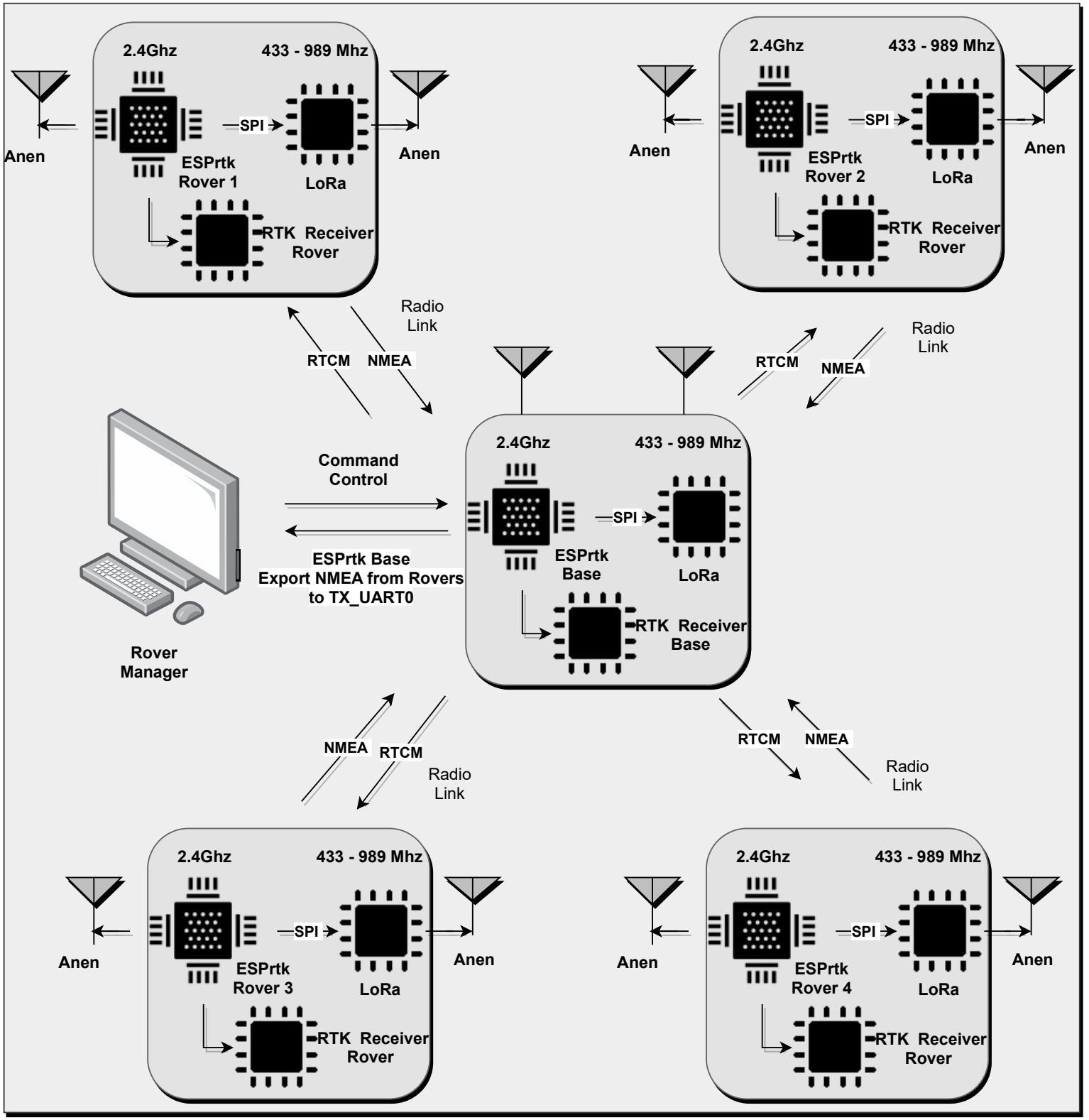



WIFI
Ethernet
Anten
SPI
ESPrtk
Rover 1
W5500
ENC28J60
RJ45
RTK Receiver
Rover
Internet
MQTT /NTRIP
Service
RTCM
NMEA
WIFI
Ethernet
Anten
SPI
ESPrtk
Rover 2
W5500
ENC28J60
RJ45
RTK Receiver
Rover
RTCM
NMEA
Internet
MQTT /NTRIP
Service
Command
Control
ESPrtk Base
Export NMEA from Rovers
to TX_UART0
Rover
Manager
WIFI
Ethernet
SPI
ESPrtk
Base
W5500
ENC28J60
RJ45
RTK Receiver
Rover
NMEA
RTCM
Internet
MQTT /NTRIP
Service
RTCM
NMEA
Internet
MQTT /NTRIP
Service
WIFI
Ethernet
Anten
SPI
ESPrtk
Rover 3
W5500
ENC28J60
RJ45
RTK Receiver
Rover
WIFI
Ethernet
Anten
SPI
ESPrtk
Rover 4
W5500
ENC28J60
RJ45
RTK Receiver
Rover

Configure Radio Link.

When the internet connection is unavailable (the NTRIP / MQTT online service is not available), ESPrtk can use its WiFi signal to transmit GNSS data.
This is similar to the full MQTT functionality but is offline. The connections auto deployed in both protocols are TCP/UDP with the **distance** constrained to the WiFi transmit power.
ESPrtk will operate stably with a 2500-3000 meters baseline when using external antennas.
For LoRa link , ESPrtk 3.0.0 support communicate to external LoRa module (SX1276 , SX1277 , SX1278 , SX12769) via SPI port for km range stransmit.
Both support real-time data encryption / decryption using AES256.

	WIFI		LoRa		Unit
	Min	Max	Min	Max	
MAC /PHY	IEEE 80211		LoRa		-
Protocol	UDP/TCP-ESPrtk ™		LoRa™ - ESPrtk™		-
AES256 Encrypt	Yes		Yes		
2 ways transmit	Yes		Yes		
Channel	1	14	1	14	-
Frequency	2.4 Ghz	2.5 Ghz	412 Mhz	915 Mhz	-
Transmit power	2	20	2	27	dBm
Transmit power	1.59	100	1.59	500	mW
Speed in the air	55	400	0.15	22	Kbps
Transmit Range	-	3	-	50	km
Bandwidth	20 Mhz	40 Mhz	56 Khz	500 Khz	-


2.4Ghz
433 - 989 Mhz
Anen
SPI
ESPrtk
Rover 1
LoRa
RTK Receiver
Rover
Radio
Link
RTCM
NMEA
ESPrtk
Rover 2
ESPrtk
Base
RTK Receiver
Base
Command
Control
ESPrtk Base
Export NMEA from Rovers
to TX_UART0
Rover
Manager
ESPrtk
Rover 3
ESPrtk
Rover 4

ESPrtk - WIFI TCP/UDP .

Act as a side action running with the main action at the same time.

This feature allows sending data get on RX_UART_RTK port (RX_UART_1 or RX_UART_2) to TCP / UDP device over WIFI connection.
Doesn't matter the format of the UART data (NMEA / RTCM / UBX / SKYTRAQ / RAW ...)
It also supports sending real-time IMU data to TCP / UDP device.
Besides, the TCP / UDP device can also act as a HOST and send control commands to ESPrtk in the same way as using on the RX / TX UART_0 port.
(HOST can send command PING, GPIO control command, check ESPrtk current system error, send restart command or control ESPrtk switch to UART / WEB Configure mode, ...
ESPrtk will respond ACK data / NACK back to HOST over TCP / UDP connection).

Supports 6 different types of connection modes and protocols:

1. TCP Server AP.
2. TCP Server STA.
3. TCP Server Client.
4. UDP Server AP.
5. UDP Server STA.
6. UDP Server Client.

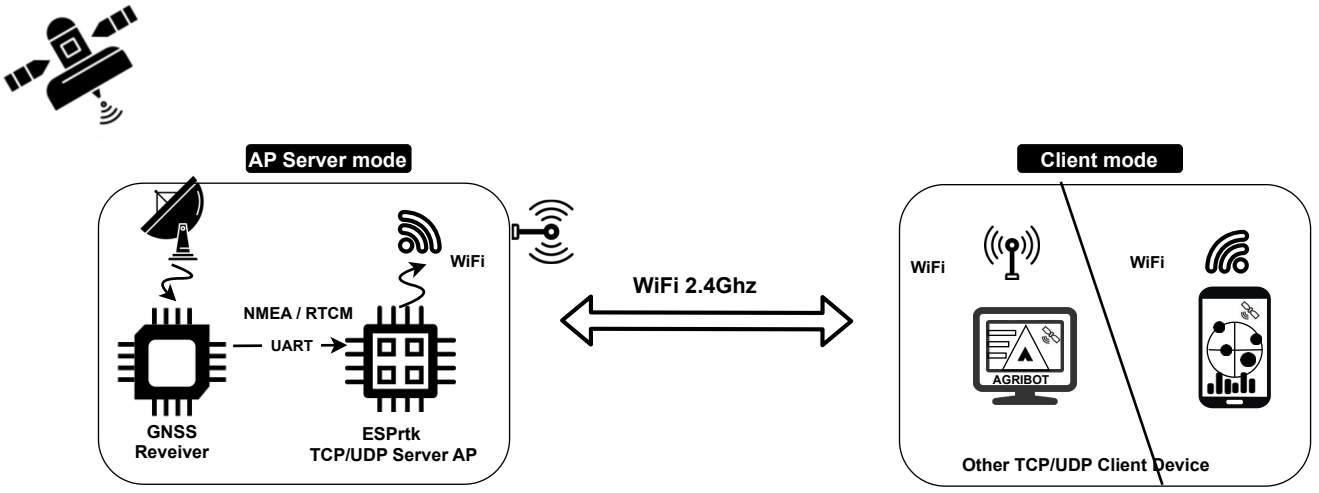
Note !

The connection mode and protocol may be different (see below), but it only serves as an intermediate data transmission channel. Usage and how ESPrtk works will be the same on all 6 connection types.

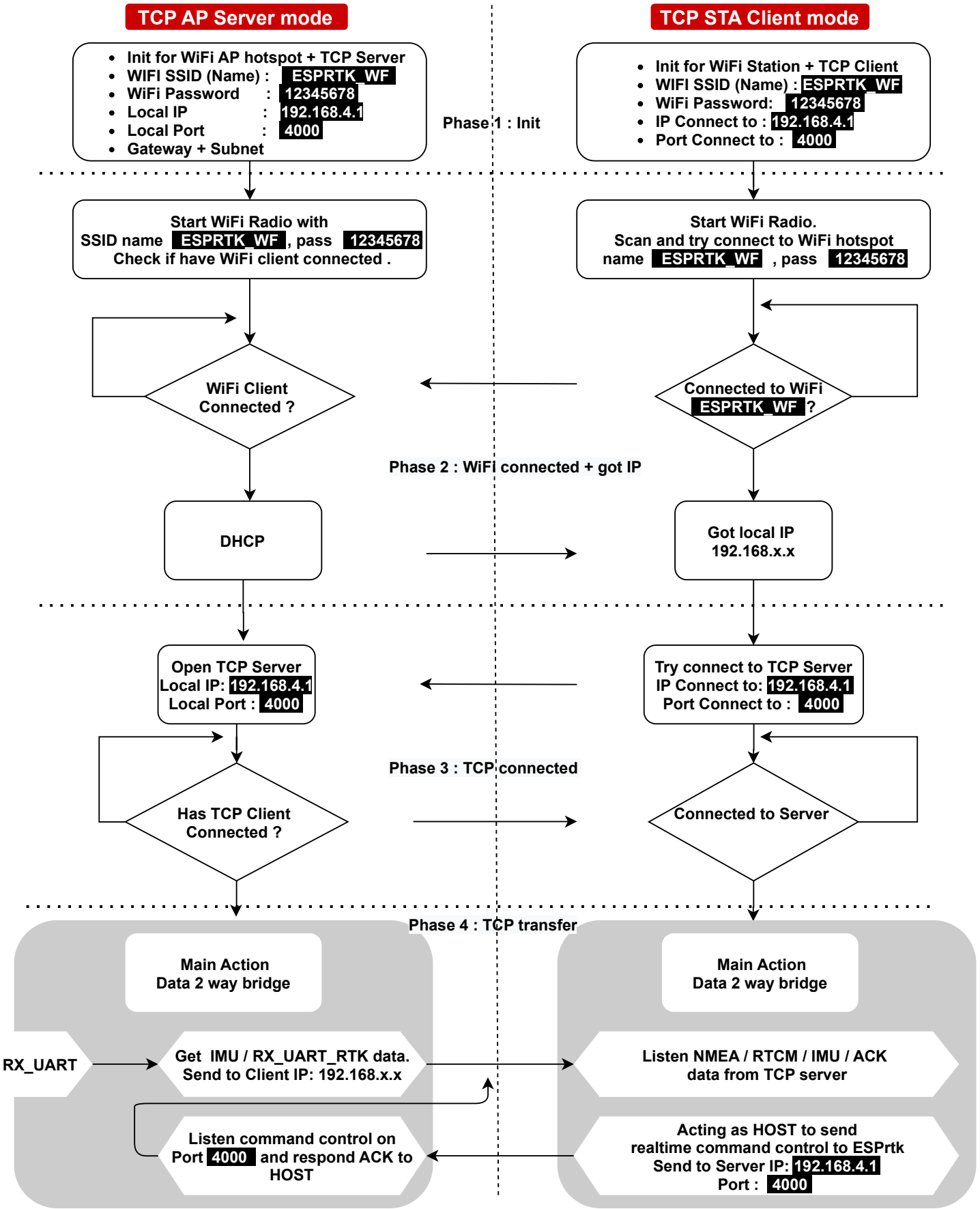
Below describes some common configurations in total of 12 different configurations.

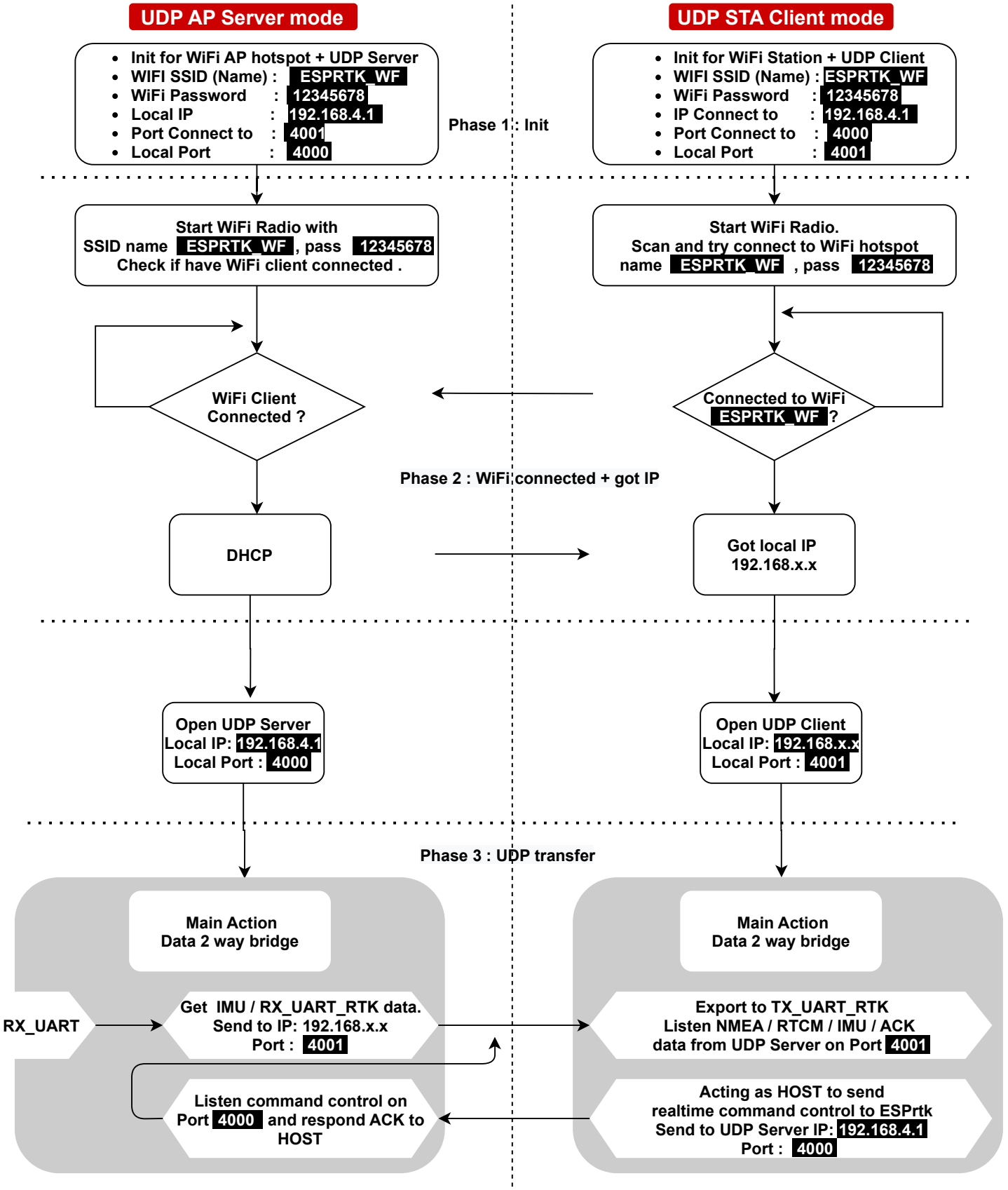
Configure ESPrtk - TCP /UDP – Server AP + Client STA mode.

Type	Detail
WiFi mode	**AP (Access Point) with STA (Station)**
Protocol	**TCP /UDP .**



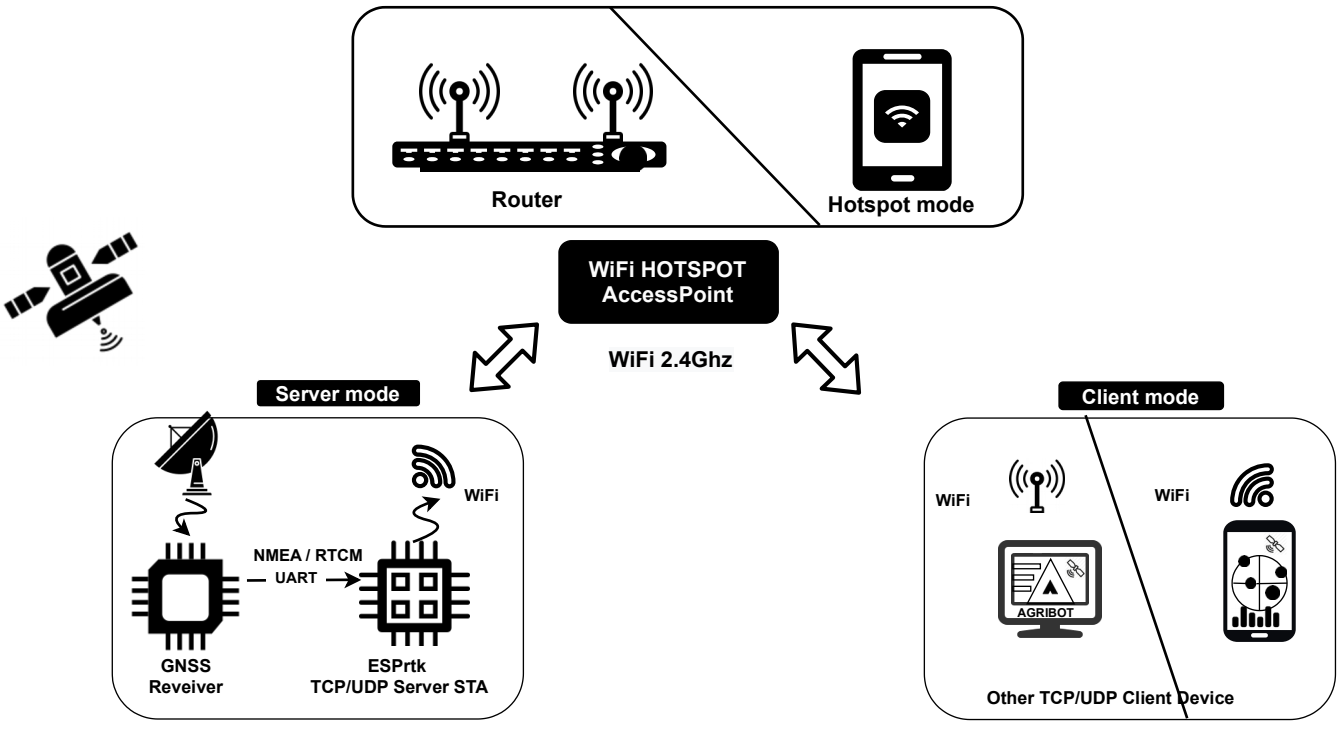
Type	WiFi mode (PHY/MAC)	Package and data Transmission Protocol	Detail
TCP/UDP Server AP	**Access Point**	**TCP/UDP Server**	Configure ESPrtk's WiFi mode as an Access Point (hotspot), wait for TCP / UDP Client device (AG monitor, Phone, ESPrtk Client ...) connect to it. Open TCP / UDP Server and listen on default IP and local port as configuration. Listens, maintains connections, and transmits data to TCP / UDP Client.
TCP/UDP Client STA	**Station**	**TCP/UDP Client**	Configure ESPrtk's WiFi mode as a Station device, try to connect to the WiFi Access Point (WiFi Access Point can be Router acting as intermediate data relay or maybe ESPrtk TCP / UDP Server AP). Open TCP / UDP Client and connect to the IP / Port of TCP / UDP Server. Maintain connectivity and data transfer to TCP / UDP Server.

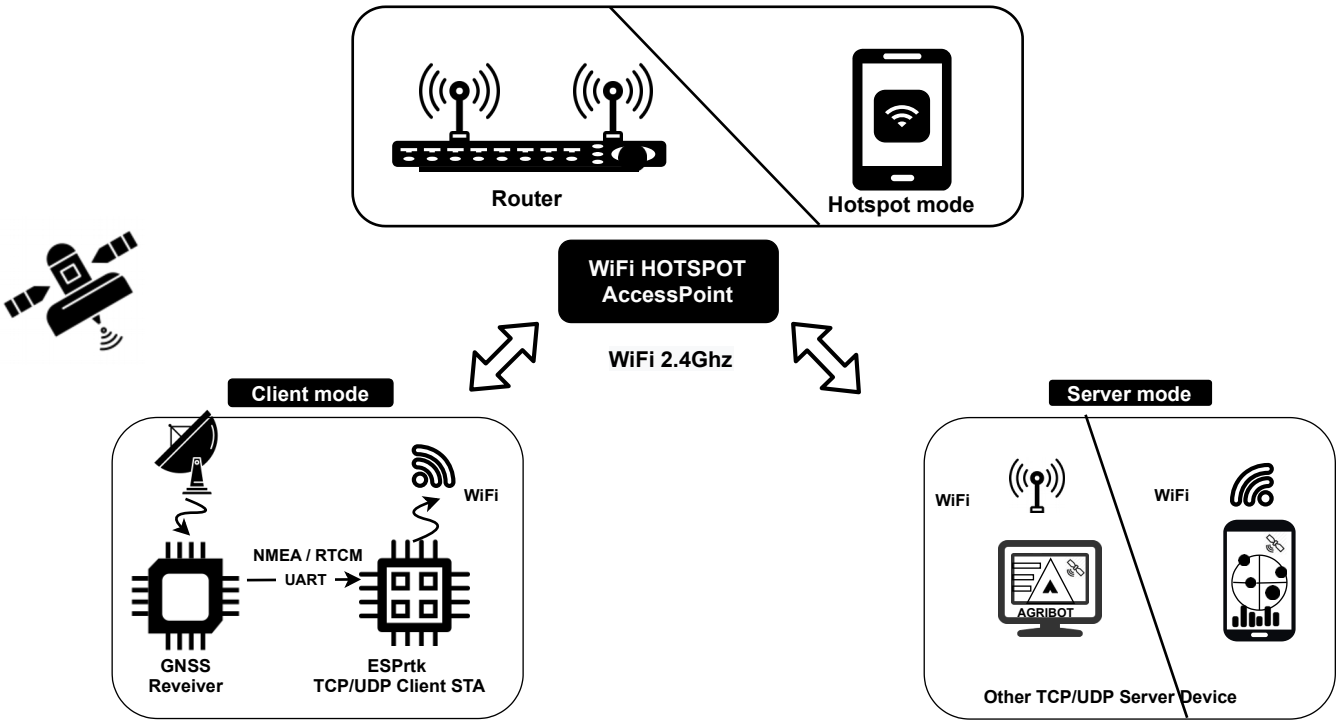

TCP AP Server mode
TCP STA Client mode
Init for WiFi AP hotspot + TCP Server
WIFI SSID (Name) : ESPRTK_WF
WiFi Password : 12345678
Local IP : 192.168.4.1
Local Port : 4000
Gateway + Subnet
Phase 1 : Init
Init for WiFi Station + TCP Client
WIFI SSID (Name) : ESPRTK_WF
WiFi Password: 12345678
IP Connect to : 192.168.4.1
Port Connect to : 4000
Start WiFi Radio with SSID name ESPRTK_WF , pass 12345678 Check if have WiFi client connected .
Start WiFi Radio. Scan and try connect to WiFi hotspot name ESPRTK_WF , pass 12345678
WiFi Client Connected ?
Connected to WiFi ESPRTK_WF ?
Phase 2 : WiFi connected + got IP
DHCP
Got local IP 192.168.x.x
Open TCP Server Local IP: 192.168.4.1 Local Port : 4000
Try connect to TCP Server IP Connect to: 192.168.4.1 Port Connect to : 4000
Phase 3 : TCP connected
Has TCP Client Connected ?
Connected to Server
Phase 4 : TCP transfer
Main Action Data 2 way bridge
Main Action Data 2 way bridge
RX_UART
Get IMU / RX_UART_RTK data. Send to Client IP: 192.168.x.x
Listen NMEA / RTCM / IMU / ACK data from TCP server
Listen command control on Port 4000 and respond ACK to HOST
Acting as HOST to send realtime command control to ESPrtk Send to Server IP: 192.168.4.1 Port : 4000


UDP AP Server mode
Init for WiFi AP hotspot + UDP Server
WIFI SSID (Name) : ESPRTK_WF
WiFi Password : 12345678
Local IP : 192.168.4.1
Port Connect to : 4001
Local Port : 4000
Phase 1 : Init
UDP STA Client mode
Init for WiFi Station + UDP Client
WIFI SSID (Name) : ESPRTK_WF
WiFi Password : 12345678
IP Connect to : 192.168.4.1
Port Connect to : 4000
Local Port : 4001
Start WiFi Radio with SSID name ESPRTK_WF , pass 12345678 Check if have WiFi client connected .
Start WiFi Radio. Scan and try connect to WiFi hotspot name ESPRTK_WF , pass 12345678
WiFi Client Connected ?
Connected to WiFi ESPRTK_WF ?
Phase 2 : WiFi connected + got IP
DHCP
Got local IP 192.168.x.x
Open UDP Server Local IP: 192.168.4.1 Local Port : 4000
Open UDP Client Local IP: 192.168.x.x Local Port : 4001
Phase 3 : UDP transfer
Main Action Data 2 way bridge
Main Action Data 2 way bridge
RX_UART
Get IMU / RX_UART_RTK data. Send to IP: 192.168.x.x Port : 4001
Export to TX_UART_RTK Listen NMEA / RTCM / IMU / ACK data from UDP Server on Port 4001
Listen command control on Port 4000 and respond ACK to HOST
Acting as HOST to send realtime command control to ESPrtk Send to UDP Server IP: 192.168.4.1 Port : 4000

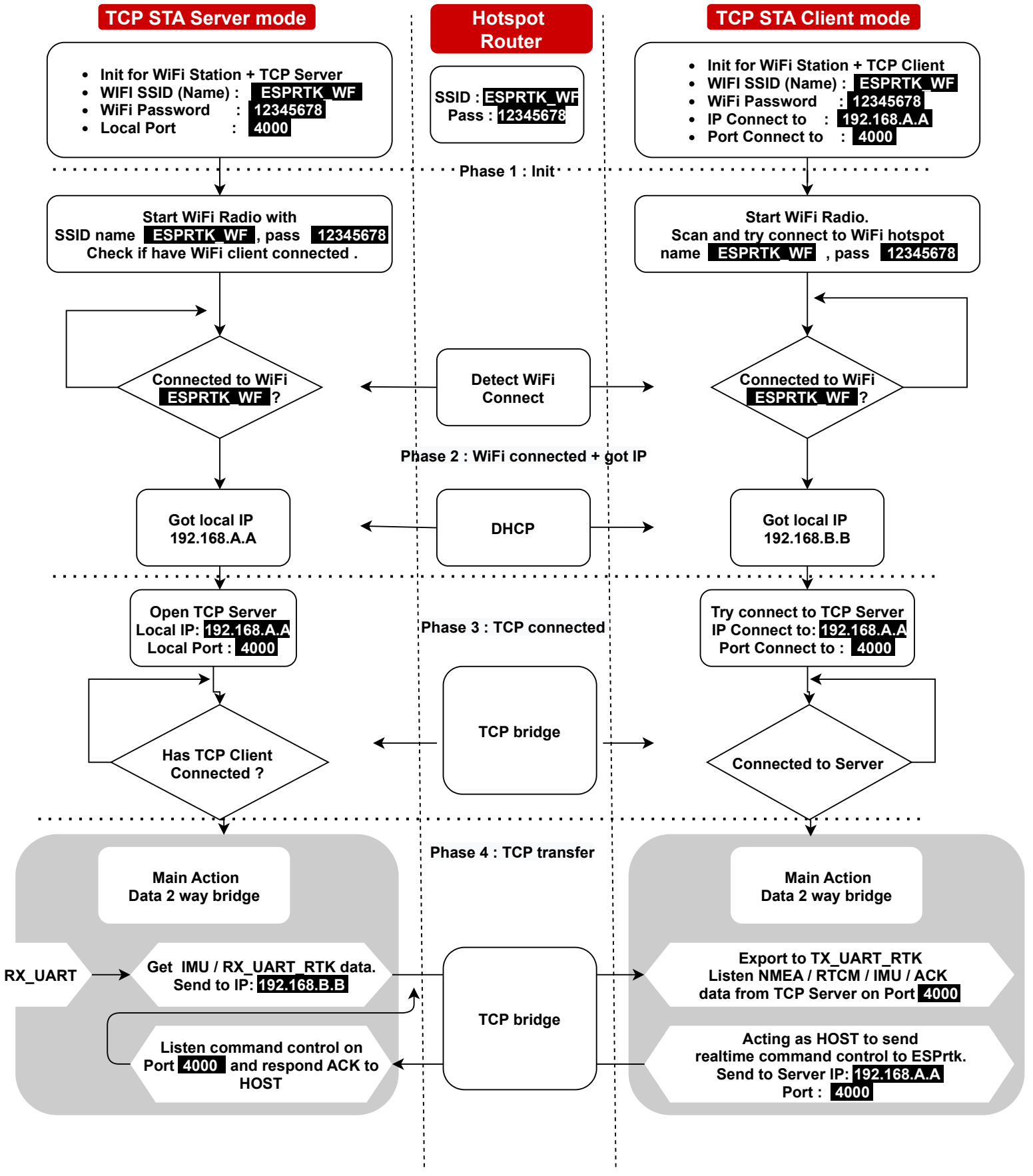
Configure ESPrtk - TCP /UDP – Server STA + Client STA mode.

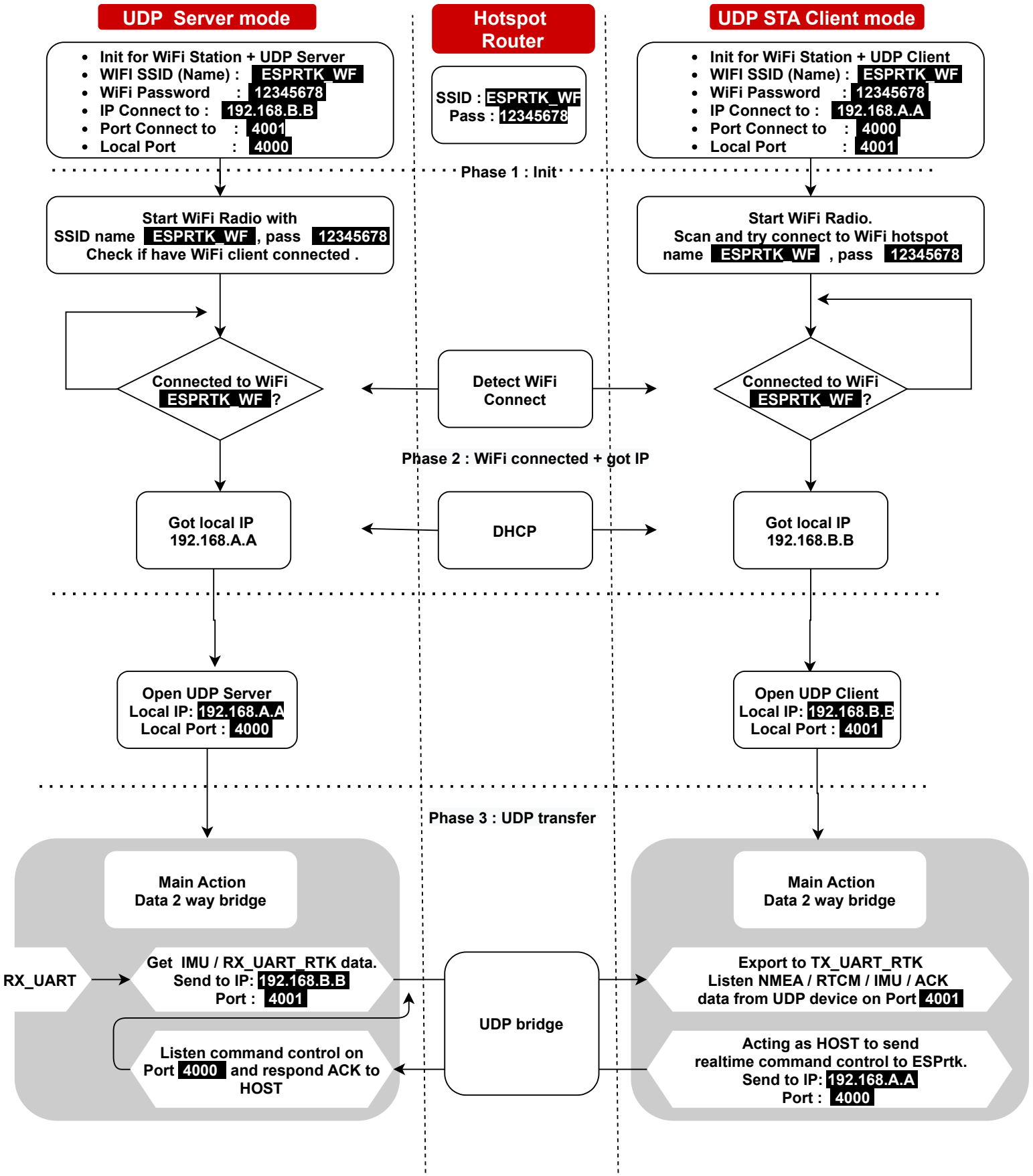
Type	Detail
WiFi mode	STA (Station) with STA (Station)
Protocol	TCP /UDP (Server / Client)





Type	WiFi mode (PHY/MAC)	Package and data Transmission Protocol	Detail
TCP/UDP Server STA	**Station**	**TCP/UDP Server**	Configure the ESPrtk's WiFi mode as a Station device, try to connect to the WiFi Access Point (Router acting as intermediate data relay). Open TCP / UDP Server and listen on default IP and Local port configuration. (Local IP is automatically assigned by the Router by DHCP). Listens, maintains connections and transmits data to TCP / UDP Client.
TCP/UDP Client STA	**Station**	**TCP/UDP Client**	Configure ESPrtk's WiFi mode as a Station device, try to connect to the WiFi Access Point (WiFi Access Point can be Router acting as intermediate data relay or maybe ESPrtk TCP / UDP Server AP). Open TCP / UDP Client and connect to the IP / Port of TCP / UDP Server. Maintain connectivity and data transfer to TCP / UDP Server.


TCP STA Server mode
Hotspot Router
TCP STA Client mode
Init for WiFi Station + TCP Server
WIFI SSID (Name) : ESPRTK_WF
WiFi Password : 12345678
Local Port : 4000
SSID : ESPRTK_WF
Pass : 12345678
Init for WiFi Station + TCP Client
WIFI SSID (Name) : ESPRTK_WF
WiFi Password : 12345678
IP Connect to : 192.168.A.A
Port Connect to : 4000
Phase 1 : Init
Start WiFi Radio with SSID name ESPRTK_WF, pass 12345678 Check if have WiFi client connected .
Start WiFi Radio. Scan and try connect to WiFi hotspot name ESPRTK_WF , pass 12345678
Connected to WiFi ESPRTK_WF ?
Detect WiFi Connect
Connected to WiFi ESPRTK_WF ?
Phase 2 : WiFi connected + got IP
Got local IP 192.168.A.A
DHCP
Got local IP 192.168.B.B
Open TCP Server Local IP: 192.168.A.A Local Port : 4000
Phase 3 : TCP connected
Try connect to TCP Server IP Connect to: 192.168.A.A Port Connect to : 4000
Has TCP Client Connected ?
TCP bridge
Connected to Server
Phase 4 : TCP transfer
Main Action Data 2 way bridge
Main Action Data 2 way bridge
RX_UART
Get IMU / RX_UART_RTK data. Send to IP: 192.168.B.B
TCP bridge
Export to TX_UART_RTK Listen NMEA / RTCM / IMU / ACK data from TCP Server on Port 4000
Listen command control on Port 4000 and respond ACK to HOST
Acting as HOST to send realtime command control to ESPrtk. Send to Server IP: 192.168.A.A Port : 4000


UDP Server mode
Hotspot Router
UDP STA Client mode
Init for WiFi Station + UDP Server
WIFI SSID (Name) : ESPRTK_WF
WiFi Password : 12345678
IP Connect to : 192.168.B.B
Port Connect to : 4001
Local Port : 4000
SSID : ESPRTK_WF
Pass : 12345678
Init for WiFi Station + UDP Client
WIFI SSID (Name) : ESPRTK_WF
WiFi Password : 12345678
IP Connect to : 192.168.A.A
Port Connect to : 4000
Local Port : 4001
Phase 1 : Init
Start WiFi Radio with SSID name ESPRTK_WF , pass 12345678 Check if have WiFi client connected .
Start WiFi Radio. Scan and try connect to WiFi hotspot name ESPRTK_WF , pass 12345678
Connected to WiFi ESPRTK_WF ?
Detect WiFi Connect
Connected to WiFi ESPRTK_WF ?
Phase 2 : WiFi connected + got IP
Got local IP 192.168.A.A
DHCP
Got local IP 192.168.B.B
Open UDP Server Local IP: 192.168.A.A Local Port : 4000
Open UDP Client Local IP: 192.168.B.B Local Port : 4001
Phase 3 : UDP transfer
Main Action Data 2 way bridge
Main Action Data 2 way bridge
RX_UART
Get IMU / RX_UART_RTK data. Send to IP: 192.168.B.B Port : 4001
UDP bridge
Export to TX_UART_RTK Listen NMEA / RTCM / IMU / ACK data from UDP device on Port 4001
Listen command control on Port 4000 and respond ACK to HOST
Acting as HOST to send realtime command control to ESPrtk. Send to IP: 192.168.A.A Port : 4000

Configure ESPrtk - NTRIP CASTER mode.

Type	Detail
WiFi mode	**STA (Server)**
Protocol	**TCP.**
Main Action	**Get RTCM/RAW /.. data from RX-UART-RTK (RX1/RX2) then send to NTRIP Client (Rover).**
	Get RTCM/RAW /.. data from NTRIP Server (BaseStation) then send to NTRIP Client (Rover).

NTRIP CASTER - Configure.

Name/Pass Login WebAdmin : For owners of NTRIP CASTER, used to login to Dashboard site to manage user accounts.

Admin Email : When NTRIP CASTER detected client error or abuse. CASTER will send an error message with the email address of the administrator (admin) via NTRIP Stream , so customers can contact them to resolve the issue.

WebAdmin Port : The port that NTRIP CASTER uses to run the Dashboard page. Usually 80.

Then , the Dashboard's address will be **http://x.x.x.x/admin** or **http://x.x.x.x:80/admin** (With x.x.x.x is the local IP address of that NTRIP CASTER) .

Setting this value to 0 will not activate the Dashboard and increase the number of sockets by 1 for the NTRIP Client / Server.

Caster Port : The port that NTRIP CASTER uses to listen to and respond to connections from the NTRIP Client (Rover / Base). Usually 2101.

Then , the Sourcetable Page's address will be **http://x.x.x.x:2101** or **http://x.x.x.x:80** (With x.x.x.x is the local IP address of that NTRIP CASTER) .

Cycle Update Parse NMEA (s) : NTRIP CASTER will collect the Rover's location via GGA / NMEA message. Setting the cycle as small will help show the location with a lower delay. Recommended value is above 60s. Setting this value to 0 disables this function.

Basestation timeout data receive (s) : NTRIP CASTER acts as a data bridge, it receives RTCM data from NTRIP Base and distributes it to all Rovers. If NTRIP Base stops sending RTCM data after a period of time, NTRIP CASTER will understand the NTRIP Base has lost connection (died or stopped working). The default value is 9 seconds.

Cycle save record limit (s) : All changes, user configuration data, bandwidth counter, etc. will be automatically saved to FLASH periodically after a period of time. The default value is 86400 seconds. (1 day). Setting this value to 0 disables this function.

Testing NTRIP CASTER with local network and dynamic IP.

Please make sure the configuration is saved correctly. Now press the “Reset” button on ESPrtk, it will start working.

NTRIP CASTER - Find the local IP address of the NTIRP CASTER.

Before starting NTRIP CASTER, the ESP32 will try to connect to the WIFI hotspot to get the IP address.

If the connection is successful, you will receive a notification with the IP address for your CASTER on the TX_UART_0 port.

If you use an OLED display for your ESP32, you will also see a message similar.

Remember, this will be the IP address that the NTRIP Client / Server will use to connect to it.

For example, in my case, the ESP32 NTRIP CASTER server now has address **192.168.1.10**

Thus, the address that the NTRIP Client needs to connect to will be:

- IP: **192.168.1.10**
- Port: **2101**

One thing to know is that this IP will be changed in the event your wifi hotspot reboots.

NTRIP CASTER - Check the status of NTRIP CASTER.

The NTRIP CASTER server is ready as soon as it connects successfully with WIFI hotspot.

Connect your computer to the same WIFI hotspot, then open the WEB browser and access the address **http://192.168.1.10:2010**

You will get a SourceTable page containing information about the active RTK Basestations (Mountpoint).

NTRIP CASTER - Connecting the F9P / PX1122R Base to the NTRIP CASTER.

By default, you will see there is a station named Base_0 in the SourceTable list, this station created by the NTRIP CASTER itself to act like an NTRIP Server / BaseStation by taking the RTCM data on its own RX_UART_RTK port and distributing it. directly to the NTIRP clients (instead of receiving the push RTCM from the NTRIP Server).

Connect TX1 pin of F9P / PX1122R Base to RX_UART_RTK (RX_UART1 or RX_UART_2) pin of ESP32 NTRIP CASTER so that this mounpoint has available data.

To connect ESPrtk with RTK receiver, See ESPrtk Pin map more detail.

NTRIP CASTER - Connect the NTRIP Client to the NTRIP CASTER.

You can use ESPrtk running in NTRIP Client mode to connect to NTRIP CASTER as an NTRIP Client / Rover.

I will use Lefeburre NTRIP CLient Adnroid app to test the connection to NTRIP CASTER. First, connect your phone to the same WIFI hotspot.

Open the application, in the NTRIP settings entry:

- Caster IP : **192.168.1.10**
- Caster Port : **2101**
- Username : **name_0**
- Password : **pass_0**
- Data Stream (Mountpoint) : **Base_0**

Tip : Mountpoint can also be automatically suggested by the application by browsing the SourceTable panel.

Then, click the connect button.

The yellow speed bar moves to the right, so I have successfully connected to Base_0 station.

Because this is NTRIP CASTER's own Base_0 station, if the F9P / PX1122R Base is not connected to the RX_UART_RTK pin of the ESP32 NTRIP CASTER, the NTRIP CASTER will only send about 50 Byte / s of default messages with a string like this in the stream. : " ST: RX_UART_RTK CASTER Not receive data after 3s ".

NTRIP CASTER - Connect the NTRIP BaseStation to the NTRIP CASTER.

You can use ESPrtk running in NTRIP Master mode to connect to NTRIP CASTER as an NTRIP Server / BaseStation.

I will use RTKLIB to test the connection to the NTRIP CASTER.

First, connect your computer to the same WIFI hotspot.

The input data is RTCM over Serial port in my case.

In NTRIP settings:

- NTRIP Caster Host : **192.168.1.10**
- Port : **2101**
- Mountpoint : **Base_1**
- Password : **pass_1**
- String : **Austin,Texas;RTCM 3.2;1005(1),1074(1),1084(1),1124(1),1230(1);;GPS+GLO+BDS;SNIP;USA;30.30;-97.73;1;0;sNTRIP;none;N;N;2980;my_misc**

Then press Start button.

The green data status bar starts moving to the right, the input data counter increases, it looks like it is working.

To be sure, I check it out again on the Sourcetable site.

And my Base_1 station appeared. So it worked.

NTRIP CASTER - Add more connections for NTRIP Client / Rover and NTRIP Server / BaseStation.

Multiple NTRIP Clients can connect and listen to RTCM data at the same time on a single mountpoint.

Multiple NTRIP Server (BaseStation) can be connected and send RTCM data to the NTRIP CASTER.

One thing to note is that the number will be limited and changed by the admin of that NTRIP CASTER.

One NTRIP account will allow up to 10 Rovers and 1 BaseStation to connect to.

An NTRIP account allows Rover to access any active Mountpoint of the NTRIP CASTER.

ESPrtk 3.9.87 supports :

- 10 different accounts
- 10 connection in the sametime
- 50000bps bandwidth for each connection

NTRIP CASTER - Admin / Dashboard page.

When the NTRIP CASTER is running, all information about the connections and the speed for each socket will be printed out on the TX_UART0 port of the ESP32.

To make it easier to manage and monitor the status of the NTRIP CASTER. ESPrtk supports an online WEB Server that works at the same time as NTRIP CASTER.

Information will be automatically updated in real time (every 1 second).

This feature is specific to Admin (manager / owner / technician) of that NTRIP CASTER.

It allows to make changes to user accounts, set bandwidth limits, uptime limits, ... Kick or ban IP on any connection, check status and location in realtime.

Can use computer, phone or any device that supports WEB browser to use with it.

NTRIP Caster Admin Dashboard - Access the Dashboard page.

First, connect your computer to the same WIFI hotspot.

(In this article, the IP address of the NTRIP CASTER is 192.168.1.10 and the Admin Dashboard page is configured to run at port 80.)

Open a browser and access the address **http://192.168.1.10:80/admin** (or **http://192.168.1.10/admin**).

The homepage appears with server type options that the browser will use to load data for the WEB site.

There are 3 options:

- **Online Server S1 Only** : Load web resources from an online server.
- **Combination Server S1 + S2** : Load web resources from online server, if failed, automatically switch to offline mode and download resource directly from ESPrtk.
- **Direct Server S2 Only** : Load web resources directly from ESPrtk.

When loading resources directly from ESPrtk (Server S2), the connection for NTRIP Client will be reduced in maximum bandwidth to ensure page loading speed. This will cause some RTCM data to be lost in about 3 seconds when the browser loads the resource from ESPrtk.

(Therefore, we recommend using the first option (Online Server S1 Only) to load the page. The 2nd and 3rd options should only be used when you are offline.)

To continue, click on the 'GO!' Button and wait about 5 seconds, a login page will appear, it asks to enter a name and password to access as admin of NTRIP CASTER, the default value is:

- Name : **admin**
- Password : **acb123ABC**

(If you want to change NTRIP CASTER admin username and password, please use the WEB configuration interface or the UART configuration interface of ESPrtk.)

NTRIP Caster Admin Dashboard - Tab Tables .

This tab provides information about the connection status for each Basestation and Rover.

General description sheet for Basestation .

It's at the top of the tab, and it provides the following information:

- **Mountpoint** : The name of the mount point on which Basestation is logged.
- **Name** : The username of the user that is used for this Basestation.

- **Active** : Whether data is running through this socket or not.
- **IP** : The IP address of that Basestation.
- **Time Run** : Total continuous operation time (from the last reconnection time).
- **RTCM In bps** : The speed at which the data is sent (up) for the Basestation. (If it is Base_0 of the NTRIP CASTER itself, it is the input data rate on RX_UART_RTK port).
- **Out bps** :Total sending speed to all rover.
- **Rovers** : Total Rover is listening to data on this mountpoint.

Detailed description sheet for Basestation.

It is located at the bottom of the general description sheet, it provides additional information:

- **STR**: STR string currently used to display on the Sourcetable page for this mount point. Position: The position (Lat / Long) of the Basestation. Click the 'Redraw map' button to show it on the map. (Make sure to access the Map tab first.)
- **Control Kick** :Instantly disconnects to this Basestation. (Basestation can still be reconnected).
- **Control Board IP** : Disconnects immediately to this Basestation and permanently ban the current IP. (Basestation cannot be reconnected by an current IP).

Detailed description sheet for Rover.

Directly below the Basestation Blueprint is a detailed description for each Rover listening to its RTCM data, which provides the following information:

- **Rover IP / Port** :The Rover IP address
- **Time Run** : Total time of continuous operation (from the last reconnection time).
- **Lasted GGA / RMC** : Complete NMEA GGA message string and Rover's Lat / Long location.
- **Position** : Rover's current position (parsed from GGA). Click the 'Redraw map' button to show it on the map. (Make sure to access the Map tab first.)
- **RTK State** : Rover's RTK FIX status (parsed from GGA).
- **NMEA In bps** : Rover's speed send NMEA/GGA to NTRIP CASTER .
- **User Acount** :Username is used for this Rover.

- **Control Kick** :Instantly disconnects to this Rover. (Rover can still be reconnected).
- **Control Board IP** : Disconnects immediately to this Rover and permanently ban the current IP. (Rover cannot be reconnected by an current IP).

IP Banned.

At the bottom of the tab is a list of banned IP addresses.

You can add, edit or remove any IP.

The number of IPs can stored on ESPrtk version 3.9.8.7 is 50 different IPs.

NTRIP Caster Admin Dashboard - Tab Home .

CASTER PROFILE Window:

- **NAME** : The owner name of this NTRIP CASTER. (Change this field using the UART Configure interface or WEB Configure in the Profile tab)
- **MAX SOCKET** : Maximum number of sockets (at the same time) that NTRIP CASTER supports
- **MAX BAN IP** : Maximum number of sockets (at the same time) that NTRIP CASTER supports
- **MAX ACCOUNT** : Number of accounts connected at the same time that NTRIP CASTER supports
- **MAX BANDWIDTH** : Maximum bandwidth NTRIP CASTER supports.
- **BASE EACH ACCOUNT** : Number of BaseStation that can share one account.
- **ROVER EACH ACCOUNT** : Number of rovers that can share one account.
- **FIRMWARE / WEB UI** :Current version of NTRIP CASTER and WEB Server interface version for Admin Dashboard site.

CURRENT TOTAL BANDWIDTH Window:

Total push-up and push-down speed of all Rover and Basestation.

TOTAL SOCKET OPENNING Window:

The total number of Rovers and Basestation is connected to.

SERVER UPTIME + STREAM COUNT BYTE Window:

- **Line 1** : Total operating time of NTRIP CASTER (from the last restart time)
- **Line 2** : Count the total number of Bytes that the NTRIP CASTER received (from the last restart).
- **Line 3** : Count the total number of Bytes that the NTRIP CASTER sent (from the last reboot).

SERVER TEMPERATURE Window:

ESP32 chip internal temperature.

Note : Please use the heatsink for the ESP32 when running in NTRIP CASTER mode for a long time.

MOST ACTIVE ACCOUNT 24 HOUR Window:

- **Top total bandwidth** : The account has the highest total number of Bytes (sending and receiving) within 24 hours.
- **Highest bandwidth** : The account has the highest bandwidth usage (receive and send) within 24 hours.
- **Top time run** : The account has the highest total connection uptime within 24 hours.
- **Top sockets** :Accounts used for the most devices within 24 hours.
- **Top average** : An account that averages the highest metrics within 24 hours.

LAST 24 HOUR REVENUE Window:

The chart depicts the highest bandwidth usage and the most connections within 24 hours.

LAST 24 MINUTES REVENUE Window:

The chart depicts the highest bandwidth usage and the most connections within 24 minutes.

MOST ACTIVE ACCOUNT 1 MINUTE Window:

- **Top total bandwidth** : The account has the highest total number of Bytes (sending and receiving) within 1 minute.
- **Highest bandwidth** : The account has the highest bandwidth usage (receive and send) within 1 minute.
- **Top time run** : The account has the highest total connection uptime within 1 minute.
- **Top sockets** :Accounts used for the most devices within 1 minute.
- **Top average** : An account that averages the highest metrics within 1 minute.

SOCKET STATUS Window:

Column chart showing the connection status and Send / Receive speed of each socket (NTRIP device).

Clicking on that column will prioritize displaying the details of that socket in the SOCKET DETAIL tab next to it.

SOCKET DETAIL Window:

Details of connection:

- **Socket** : Connection's socket ID.
- **Mode** : Connection mode (Base / Rover).
- **IP** : IP and Port of Rover/Base.
- **Time** : Total time to maintain the connection (from the last reconnection time).
- **Print** : The speed of the input data on this socket.
- **Out** : The speed of the output data on this socket.

- **In + Out** : The total data rate of input and output on this socket.
- **Lat** : Rover/Base's latitude .
- **Long** : Rover/Base's longitude.
- **Account** :The account that this Rover/Base is using to log in for this connection.
- **Agent** : NTRIP user agent.

BANDWIDTH Window:

- **ALL** : Total input and output data rate of the NTRIP CASTER at the present time.
- **IN** : Total input rate of NTRIP CASTER at the present time.
- **OUT** : The total output rate of the NTRIP CASTER at the present time.

SOCKET Window:

- **ALL** : Total number of devices connected to NTRIP CASTER (Base + Rover) at the present time.
- **Base** : The total number of devices connected to the NTRIP CASTER as a Base at the present time.
- **Rover** : Total number of devices connected to the NTRIP CASTER as a Rover at the present time.

NTRIP Caster Admin Dashboard - Tab User.

User name/Login name : Username for NTRIP Client account.

Password : Password for NTRIP Client / Server account

Mountpoint :Registered Mountpoint name for NTRIP Server (BaseStation) account.

Rover Support : Number of rovers sharing one account.

VRS Support: VRS support Flag . Tick select to enable it.

STR :

- STR editor displayed on Sourcetable for NTRIP Server (BaseStation) account.
- Setting the flag 'Enable use STR from Client' to TRUE will display the STR uploaded from the NTRIP Server (BaseStation) account set by the user.

- Setting the flag 'Enable use STR from Client' to FLASE will display the default STR set by the admin.

Highest Bandwidth :

- Set the highest speed limit for this account. If the speed counter exceeds this limit, the account will be permanently banned until the counter is cleared by the admin.
- Unit : bps (bit-per-second).
- If this account is used on more than one Rover at a time, the speed value will include the total speeds of all Rovers.
- Setting the value to 0 will disables this function.

Max Total Bandwidth:

- Set a limit for the total number of Bytes (In + Out) for this account.
- If the Byte count counter exceeds this limit, the account will be permanently banned until the counter is cleared by the admin.
- Unit : GB (Giga Byte).
- If this account is used on multiple Rovers at the same time, the Byte counter will include the total Bytes of all Rovers.
- Setting this value to 0 will disables this function.

Max Total Time :

- Set a time limit for this account.
- If the timer exceeds this limit, the account will be permanently banned until the timer is cleard by the admin.
- Unit : Second.
- If this account is used on multiple Rovers at the same time, the timer will include the total times of all Rovers.
- Setting this value to 0 will disables this function.

Save the configure :

- When doing any update, add new, edit, delete, ... any data field. All information will be stored in the RAM (temporary) of ESPrtk.
- A button will appear at the top of the Web page, clicking on it to save all changes to the ESPrtk FLASH memory.
- It may take more than 3 seconds to save to FLASH, which may cause the NTRIP connection to be interrupted, so be sure to only perform saving the configuration to FLASH when you are sure.

NTRIP Caster Admin Dashboard - Tab Map.

- Display all locations of NTRIP Rover and NTRIP Basestation on a single page.
- Clicking on the Marker will display a Pop-up detailing the NTRIP Rover / Basestation information.
- If a location cannot be found for that NTRIP account, the marker will be set to the default coordinate (0,0).

NTRIP Caster Admin Dashboard - Tab Admin.

- **Logout** : Log out of the Dashboard page.
- **Kick All Socket** : Disconnects all active NTRIP accounts.
- **Restart Caster** : Restart the NTRIP CASTER remotely.

Configure ESPrtk - UART_to_WIFI_Bridge mode

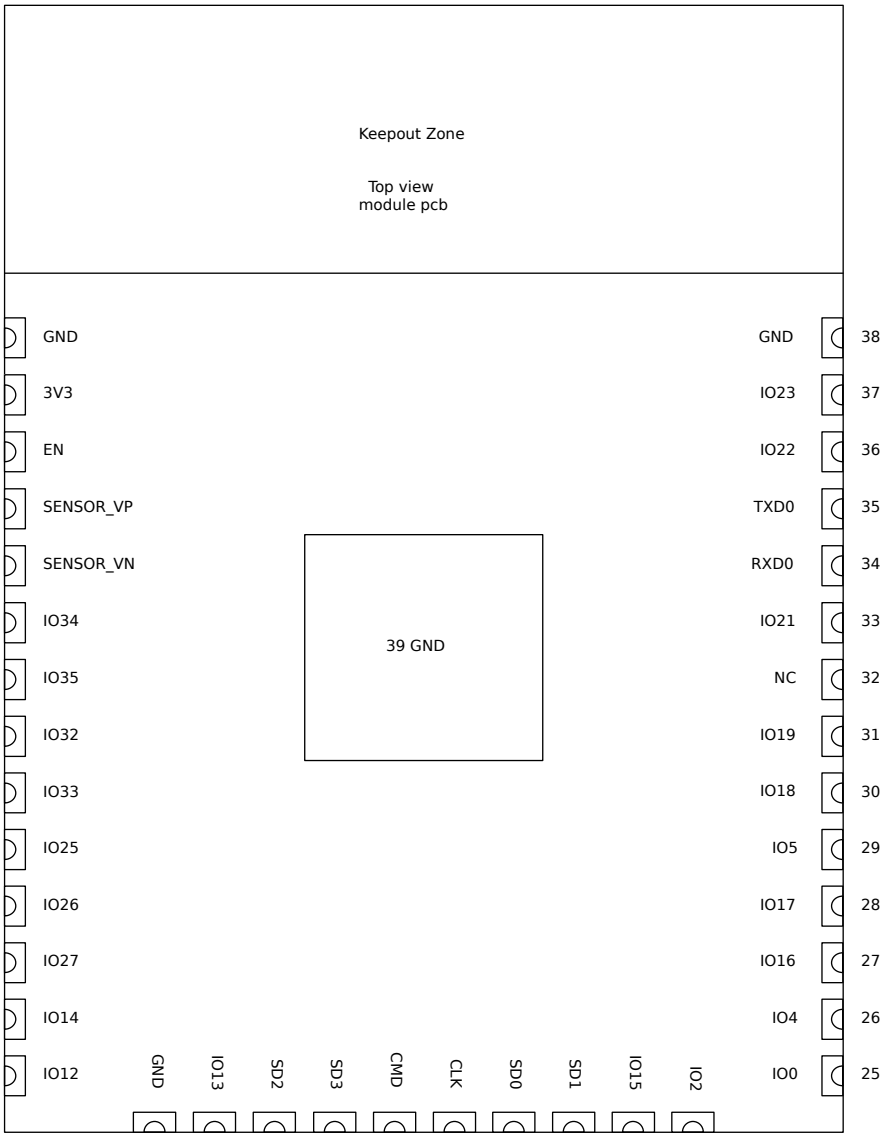
Main Action	Detail
UART_to_WIFI_Bridge	• Send data (from RX-UART-RTK) to (WIFI TCP/UDP Device) • + Export data (from WIFI TCP/UDP Device) to TX-UART-RTK port as RTK Rover

Other Action for UART_to_WIFI_Bridge	UART_to_WIFI_Bridge
Export IMU Data to TX_UART_0 port	●
Save NMEA/RTCM/RAW/.. Data (from RX-UART-RTK) to SD Card	●
Send NMEA/RTCM/RAW/.. Data (from RX-UART-RTK) to Bluetooth Client	●
Send IMU_1 Data to Bluetooth Client	●
Send IMU_2 Data to Bluetooth Client	●
Send Data (from RX-UART-RTK) to WiFi TCP/UDP device	●
Send IMU_1 Data to WiFi TCP/UDP device	●
Send IMU_2 Data to WiFi TCP/UDP device	●
Respond realtime command control on Bluetooth Client	●
Respond realtime command control on WiFi TCP/UDP	●

ESP32 Pin Definitions

ESP32 module

Pin Layout



Strapping Pins

ESP32 has five strapping pins, which can be seen in Chapter 6 Schematics:
• MTDI – GPIO12.
• GPIO0
• GPIO2
• MTDO - GPIO15.
• GPIO5
Software can read the values of these five bits from register ”GPIO_STRAPPING”.
During the chip’s system reset release (power-on-reset, RTC watchdog reset and brownout reset), the latches of the strapping pins sample the voltage level as strapping bits of ”0” or ”1”, and hold these bits until the chip is powered down or shut down. The strapping bits configure the device’s boot mode, the operating voltage of VDD_SDIO and other initial system settings.
Each strapping pin is connected to its internal pull-up/pull-down during the chip reset. Consequently, if a strapping pin is unconnected or the connected external circuit is high-impedance, the internal weak pull-up/pull-down will determine the default input level of the strapping pins.
To change the strapping bit values, users can apply the external pull-down/pull-up resistances, or use the host MCU’s GPIOs to control the voltage level of these pins when powering on ESP32.
After reset release, the strapping pins work as normal-function pins.
Refer to Table 4 for a detailed boot-mode configuration by strapping pins.

Booting Mode					
Pin	Default	SPI Boot		Download Boot	
GPIO0	Pull-up	1		0	
GPIO2	Pull-down	Don’t-care		0	
Voltage of Internal LDO (VDD_SDIO)					
Pin	Default	3.3 V		1.8 V	
MTDI - GPIO12	Pull-down				
Enabling/Disabling Debugging Log Print over U0TXD During Booting					
Pin	Default	U0TXD Active		U0TXD Silent	
MTDO - GPIO15	Pull-up	1		0	
Timing of SDIO Slave					
Pin	Default	Falling-edge Sampling Falling-edge Output	Falling-edge Sampling Rising-edge Output	Rising-edge Sampling Falling-edge Output	Rising-edge Sampling Rising-edge Output
MTDO - GPIO15	Pull-up	0	0	1	1

GPIO5	Pull-up	0	1	0	1

Note :
• Firmware can configure register bits to change the settings of ”Voltage of Internal LDO (VDD_SDIO)” and ”Timing
of SDIO Slave” after booting.
• Both ESP32-WROOM-32D and ESP32-WROOM-32U integrate a 3.3 V SPI flash, so the pin MTDI cannot be set to 1 when the modules are powered up.

Electrical Characteristics

Absolute Maximum Ratings

Stresses beyond the absolute maximum ratings listed in Table 5 below may cause permanent damage to the device. These are stress ratings only, and do not refer to the functional operation of the device that should follow the recommended operating conditions.

Symbol	Parameter	Min	Max	Unit
VDD33	Power supply voltage	–0.3	3.6	V
I_{output} [1]	Cumulative IO output current	-	[1,100]	mA
T_{store}	Storage temperature	–40	150	°C

1. The module worked properly after a 24-hour test in ambient temperature at 25 °C, and the IOs in three domains (VDD3P3_RTC, VDD3P3_CPU, VDD_SDIO) output high logic level to ground. Please note that pins occupied by flash and/or PSRAM in the VDD_SDIO power domain were excluded from the test.
2. Please see Appendix IO_MUX of ESP32 Datasheet for IO’s power domain.

Recommended Operating Conditions

Symbol	Parameter	Min	Typical	Max	Unit
VDD33	Power supply voltage	3.0	3.3	3.6	V
I_{VDD}	Current delivered by external power supply	0.5	-	-	A
T	Operating temperature	–40	-	85	°C

DC Characteristics (3.3 V, 25 °C)

Symbol	Parameter		Min	Typ	Max	Unit
C_IN	Pin capacitance		-	2	-	pF
V_IH	High-level input voltage		0.75×VDD	-	VDD1+0.3	V
V_IL	Low-level input voltage		–0.3	-	0.25×VDD note1	V
I_IH	High-level input current		-	-	50	nA
I_IL	Low-level input current		-	-	50	nA
V_OH	High-level output voltage		0.8×VDD1	-	-	V
V_OL	Low-level output voltage		-	-	0.1×VDD note1	V
I_OH	High-level source current (VDD1 = 3.3 V, VOH >= 2.64 V output drive strength set to the maximum)	VDD3P3_CPU power domain note2	-	40	-	mA
		VDD3P3_RTC power domain note2	-	40	-	mA
		VDD_SDIO power domain note3	-	20	-	mA
I_OL	Low-level sink current (VDD1 = 3.3 V, VOL = 0.495 V, output drive strength set to the maximum)		-	28	-	mA
R_PU	Resistance of internal pull-up resistor		-	45	-	kΩ
R_PD	Resistance of internal pull-down resistor		-	45	-	kΩ
V_IL_nRST	Low-level input voltage of CHIP_PU to power off the chip		-	-	0.6	V

Notes:
1. Please see Appendix IO_MUX of **ESP32 Datasheet** for IO's power domain. VDD is the I/O voltage for a particular power domain of pins.
2. For VDD3P3_CPU and VDD3P3_RTC power domain, per-pin current sourced in the same domain is gradually reduced from around 40 mA to around 29 mA, VOH>=2.64 V, as the number of current-source pins increases.
3. Pins occupied by flash and/or PSRAM in the VDD_SDIO power domain were excluded from the test.

Wi-Fi Radio

Parameter	Condition	Min	Typical	Max	Unit
Operating frequency range note1	-	2412		2484	MHz
Output impedance note2	-	-	note 2	-	Ω

TX power [note3]	11n, MCS7	12	13	14	dBm
	11b mode	17.5	18.5	20	dBm
	11b, 1 Mbps	-	-98	-	dBm
	11b, 11 Mbp	-	-89	-	dBm
	11g, 6 Mbps	-	-92	-	dBm
	11g, 54 Mbps	-	-74	-	dBm
	11n, HT20, MCS0	-	-91	-	dBm
	11n, HT20, MCS7	-	-71	-	dBm
	11n, HT40, MCS0	-	-89	-	dBm
	11n, HT40, MCS7	-	-69	-	dBm
Adjacent channel rejection	11g, 6 Mbps	-	31	-	dB
	11g, 54 Mbps	-	14	-	dB
	11n, HT20, MCS0	-	31	-	dB
	11n, HT20, MCS7	-	13	-	dB

Note 1. Device should operate in the frequency range allocated by regional regulatory authorities. Target operating frequency
range is configurable by software.
Note 2. For the modules that use IPEX antennas, the output impedance is 50 Ω. For other modules without IPEX antennas,
users do not need to concern about the output impedance.
Note 3. Target TX power is configurable based on device or certification requirements.

BLE Radio Receiver

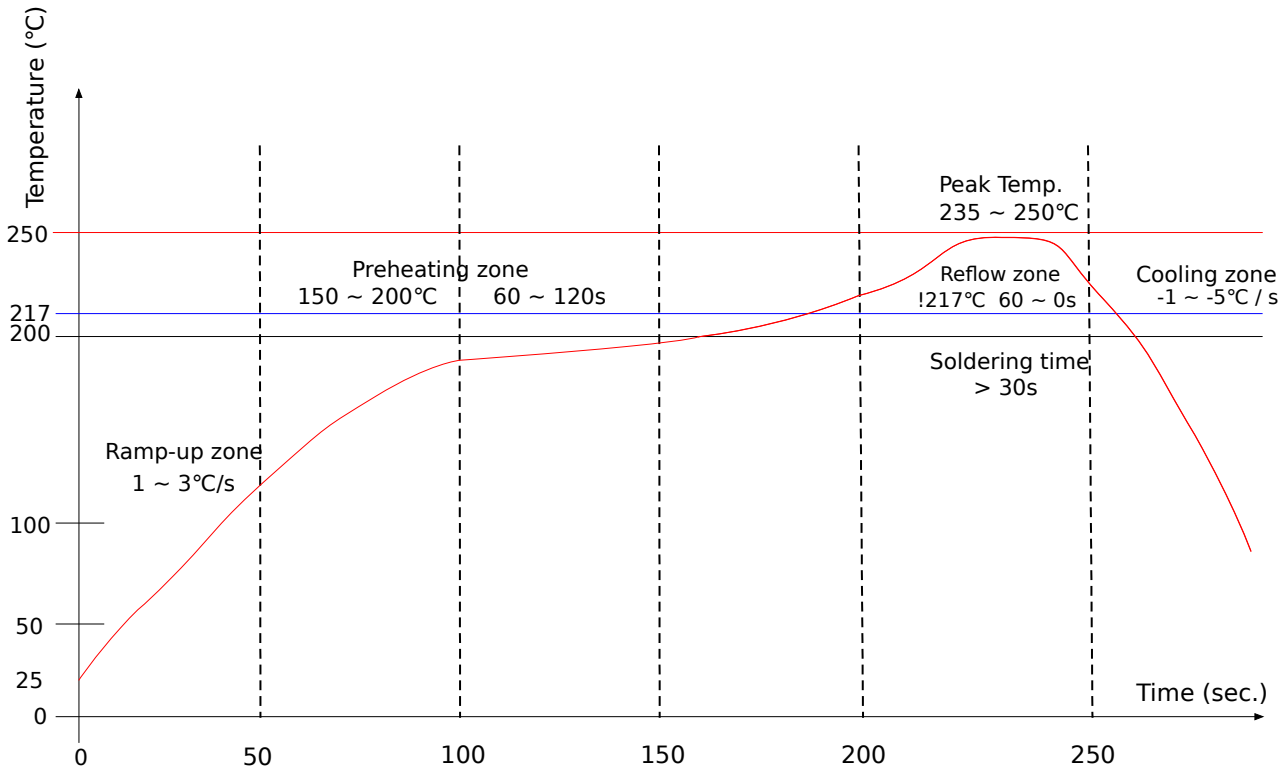
Parameter	Conditions	Min	Typ	Max	Unit
Sensitivity @30.8% PER	-	-	–97		dBm
Maximum received signal @30.8% PER	-	0	-		dBm
Co-channel C/I	-	-	+10		dB
Adjacent channel selectivity C/I	F = F0 + 1 MHz	-	-5		dB
	F = F0 – 1 MHz	-	-5		dB
	F = F0 + 2 MHz	-	-25		dB
	F = F0 – 2 MHz	-	-35		dB
	F = F0 + 3 MHz	-	-25		dB

	F = F0 – 3 MHz	-	-45		dB
Out-of-band blocking performance	30 MHz ~ 2000 MHz	–10			dBm
	2000 MHz ~ 2400 MHz	–27			dBm
	2500 MHz ~ 3000 MHz	–27			dBm
	3000 MHz ~ 12.5 GHz	–10			dBm
	-	–36			dBm

BLE Radio Transmitter .

Parameter	Conditions	Min	Typ	Max	Unit
RF transmit power	-	-	0		dBm
Gain control step	-	-	3		dBm
RF power control range	-	–12	-	+9	dBm
Adjacent channel transmit power	F = F0 ± 2 MHz	-	-52		dBm
	F = F0 ± 3 MHz	-	-58		dBm
	F = F0 ± > 3 MHz	-	-60		dBm
Δf_1_{avg}	-	-	-	265	kHz
Δf_2_{max}	-	247	-		kHz
Δf_1_{avg} / Δf_1_{avg}	-	-	-0.92		-
ICFT	-	-	-10		kHz
Drift rate	-	-	0.7		kHz/ 50 µs
Drift	-	-	2		kHz

Reflow Profile



Ramp-up zone — Temp.: <150°C Time: 60 ~ 90s Ramp-up rate: 1 ~ 3°C/s
Preheating zone — Temp.: 150 ~ 200°C Time: 60 ~ 120s Ramp-up rate: 0.3 ~ 0.8°C/s
Reflow zone — Temp.: >217°C 7LPH60 ~ 90s; Peak Temp.: 235 ~ 250°C (<245°C recommended) Time: 30 ~ 70s

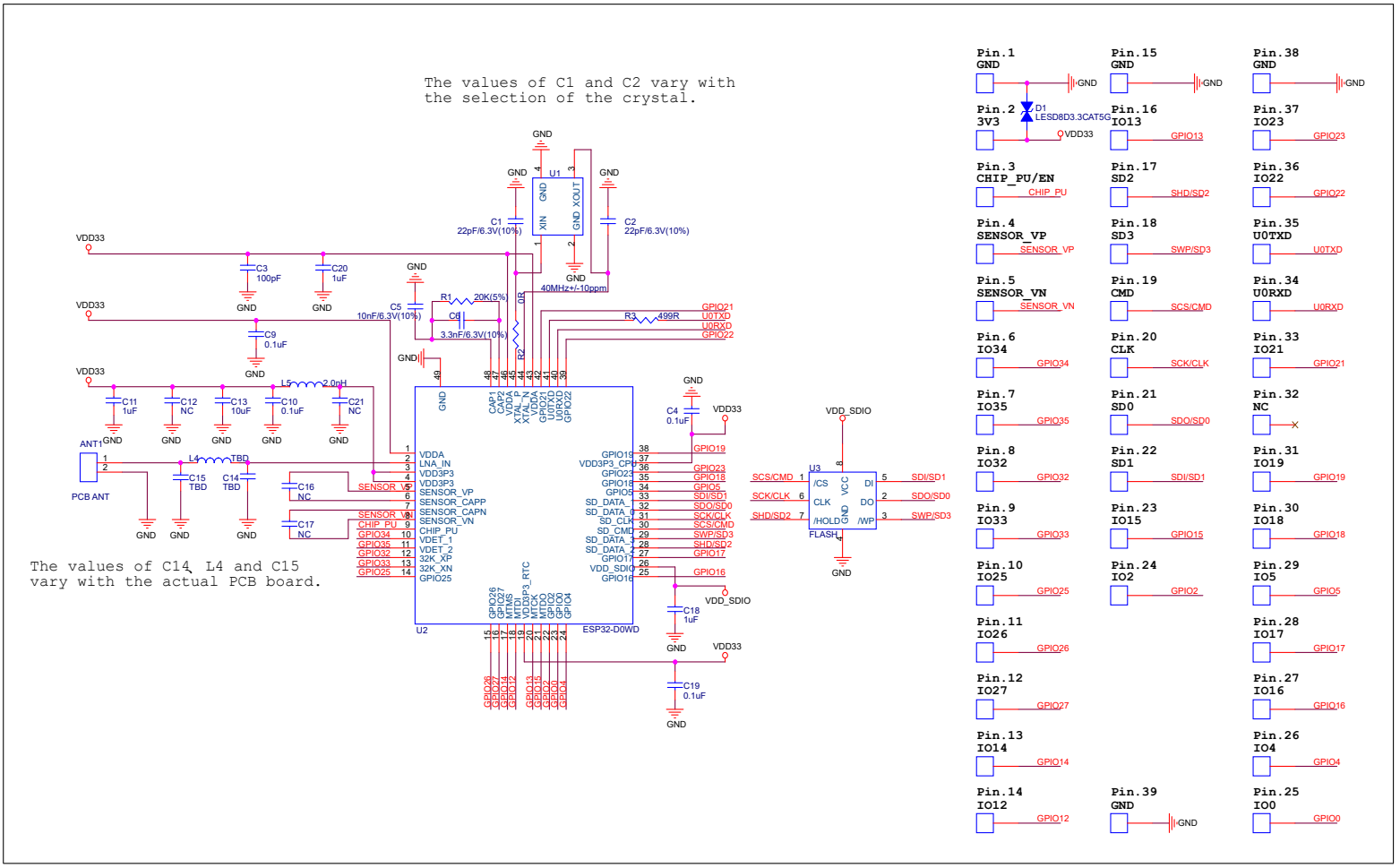
ESP32-WROOM-32D/E Schematics

Note:

• Soldering Pad 39 to the Ground is not necessary for a satisfactory thermal performance. If users do want to solder it, they need to ensure that the correct quantity of soldering paste is applied. • When ESP32 is powered on and off repeatedly by switching the power rails, and there is a large capacitor on the 3V3 rail, a discharge circuit can be added to the 3V3 rail to ensure proper power-on-reset. Please find the discharge circuit in Chapter Peripheral Schematics, in ESP32-WROOM-32 Datasheet.

• When battery is used as the power supply for ESP32 series of chips and modules, a supply voltage supervisor is recommended to avoid boot failure due to low voltage. Users are recommended to pull CHIP_PU low if the power supply for ESP32 is below 2.3 V. For the reset circuit, please refer to Chapter Peripheral Schematics, in ESP32-WROOM-32 Datasheet.

• To ensure the power supply to the ESP32 chip during power-up, it is advised to add an RC delay circuit at the EN pin. The recommended setting for the RC delay circuit is usually R = 10 kΩ and C = 0.1 µF. However, specific parameters should be adjusted based on the power-up timing of the module and the power-up and reset sequence timing of the chip. For ESP32's power-up and reset sequence timing diagram, please refer to Section Power Scheme in ESP32 Datasheet.



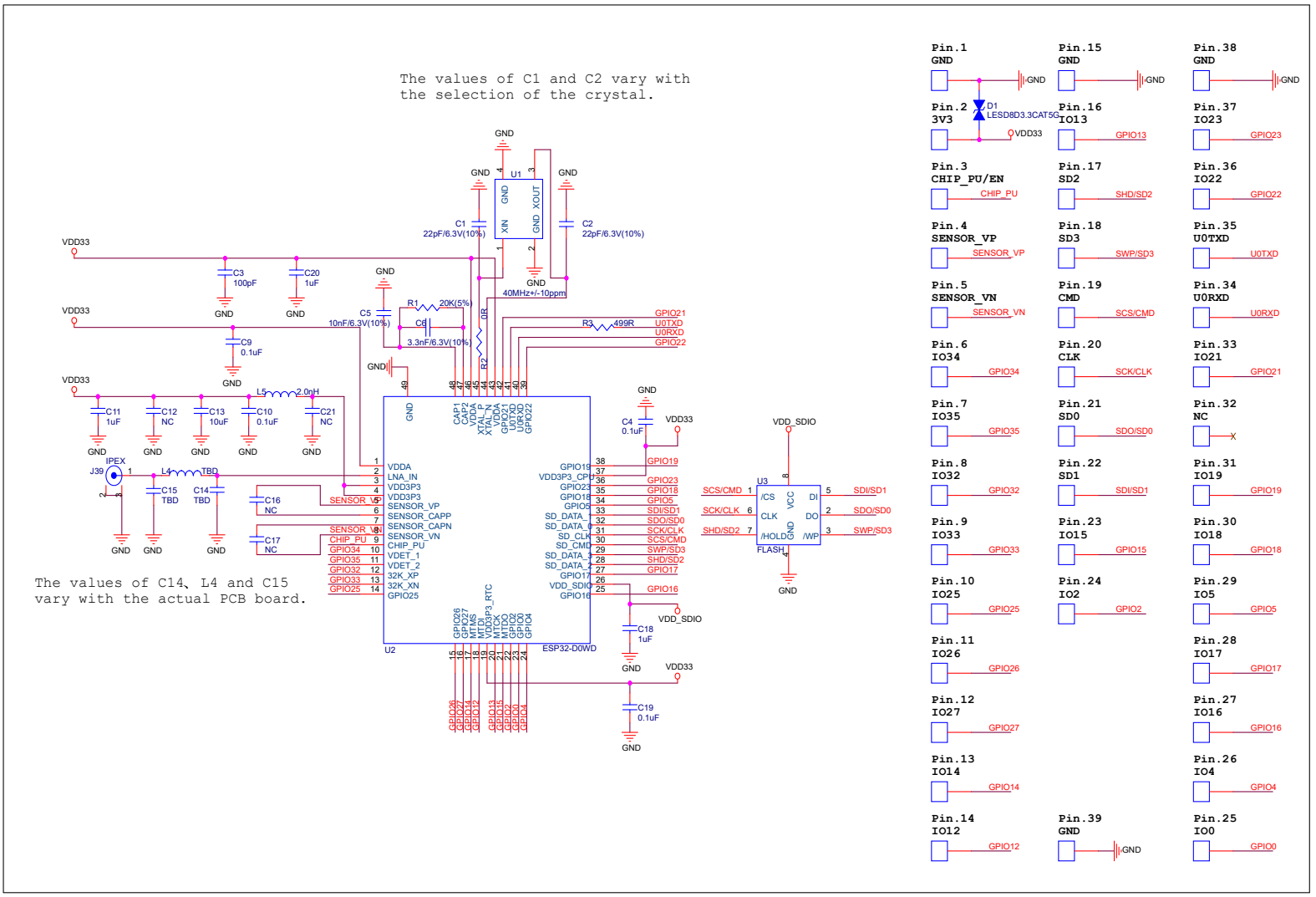
ESP32-WROOM-32U/UE Schematics

Note:

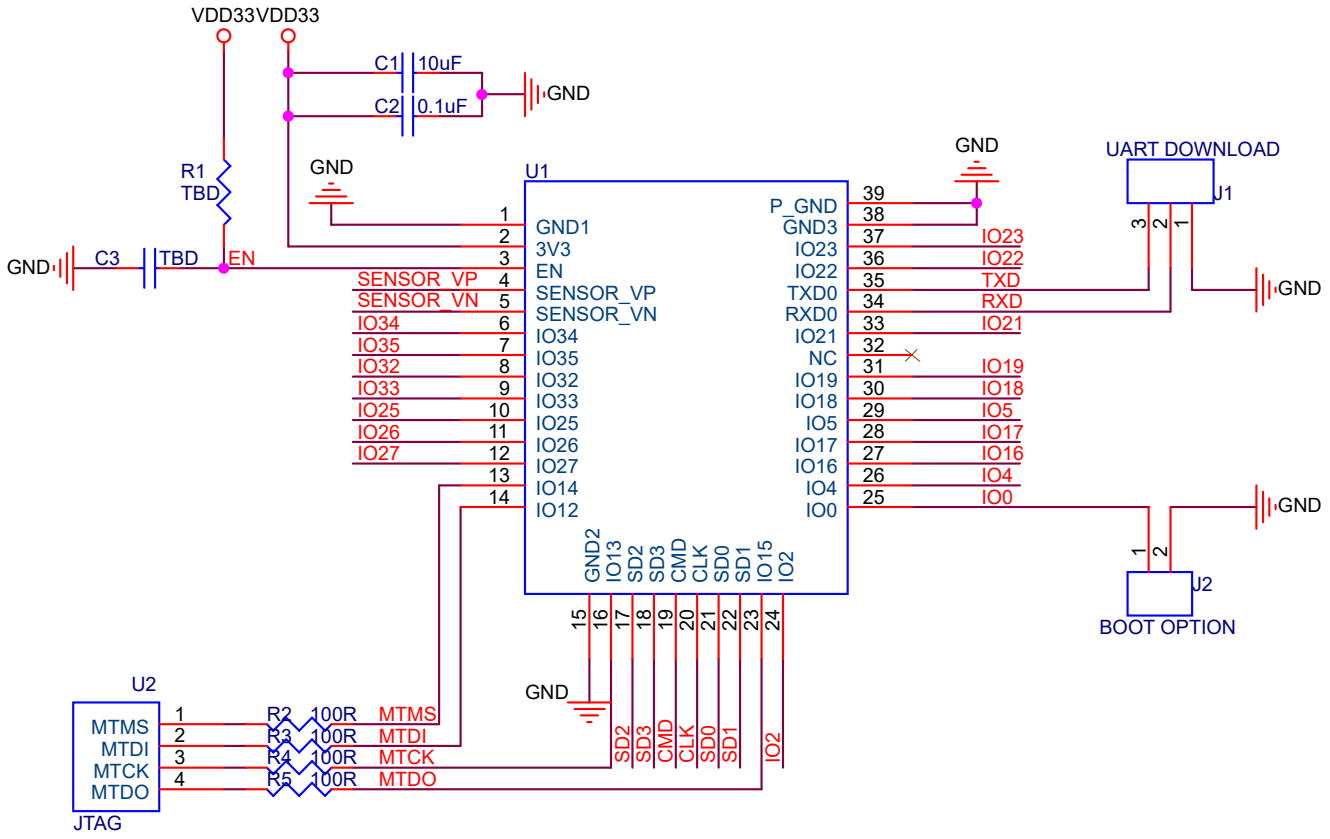
• Soldering Pad 39 to the Ground is not necessary for a satisfactory thermal performance. If users do want to solder it, they need to ensure that the correct quantity of soldering paste is applied. • When ESP32 is powered on and off repeatedly by switching the power rails, and there is a large capacitor on the 3V3 rail, a discharge circuit can be added to the 3V3 rail to ensure proper power-on-reset. Please find the discharge circuit in Chapter Peripheral Schematics, in ESP32-WROOM-32 Datasheet.

• When battery is used as the power supply for ESP32 series of chips and modules, a supply voltage supervisor is recommended to avoid boot failure due to low voltage. Users are recommended to pull CHIP_PU low if the power supply for ESP32 is below 2.3 V. For the reset circuit, please refer to Chapter Peripheral Schematics, in ESP32-WROOM-32 Datasheet.

• To ensure the power supply to the ESP32 chip during power-up, it is advised to add an RC delay circuit at the EN pin. The recommended setting for the RC delay circuit is usually R = 10 kΩ and C = 0.1 µF. However, specific parameters should be adjusted based on the power-up timing of the module and the power-up and reset sequence timing of the chip. For ESP32's power-up and reset sequence timing diagram, please refer to Section Power Scheme in ESP32 Datasheet.



Peripheral Schematics



MTDI should be kept at a low electric level when powering up the module.

ESP32-WROOM-32D/E & ESP32-WROOM-32U/UE Peripheral Schematics

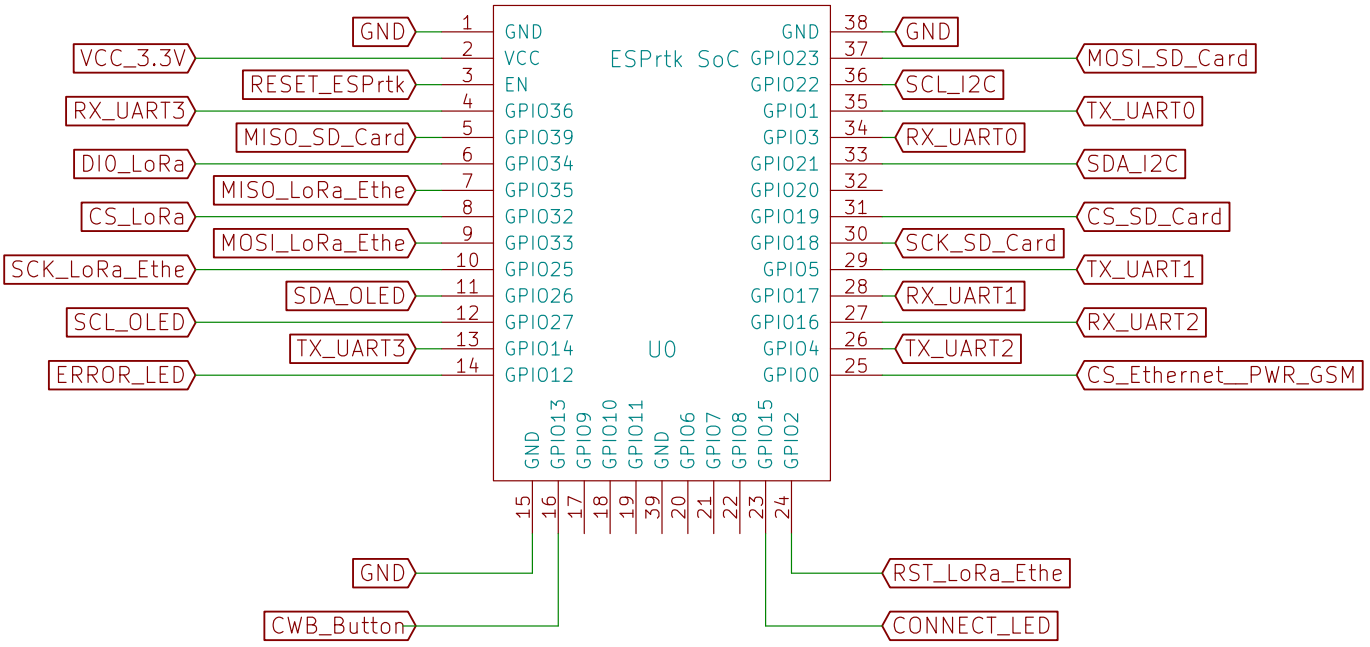
ESPrtk PinMap

ESPrtk IO Name	GPIO	ESPrtk IO Name	GPIO	ESPrtk IO Name	GPIO
CWB_Button	13	RX_UART0	3	MOSI_SD_Card	23
MOSI_LoRa_Ethe	33	TX_UART0	1	MISO_SD_Card	39
MISO_LoRa_Ethe	35	RX_UART1	17	SCK_SD_Card	18
SCK_LoRa_Ethe	25	TX_UART1	5	CONNECT_LED	15
CS_SD_Card	19	RX_UART2	16	ERROR_LED	12
CS_LoRa	32	TX_UART2	4	TX_UART3	14
CS_Ethernet__PWR_GSM	0	SCL_I2C	22	SCL_OLED	27
RST_LoRa_Ethe	2	SDA_I2C	21	SDA_OLED	26
DI0_LoRa	34	RX_UART3	36		

ESPrtk IO Name	GPIO	Input Pullup	Out	Type	Function
VCC_3.3V	VCC	●	○	○	Power input
GND	GND	○	●	○	Power ground
RESET_ESPrtk	EN	●	○	Inerrupt	Reset ESPrtk
ESPrtk IO Name	**GPIO**	**Input Pullup**	**Out**	**Type**	**Function**
CWB_Button	13	●	○	Inerrupt	Hold to Start WEB Configure
MOSI_LoRa_Ethe	33	○	●	SPI	Connect to MOSI pin on both LoRa and Ethernet module
MISO_LoRa_Ethe	35	●	○	SPI	Connect to MISO pin on both LoRa and Ethernet module
SCK_LoRa_Ethe	25	○	●	SPI	Connect to SCK pin on both LoRa and Ethernet module
CS_SD_Card	19	○	●	SPI	Connect to CS pin on SD /Micro SD Card module
CS_LoRa	32	○	●	SPI	Connect to CS pin on LoRa module
CS_Ethernet__PWR_GSM	0	○	●	SPI	Connect to CS pin on Ethernet module or Connect to PWR (Power KEY ON/OFF) of GSM/Cellular module
RST_LoRa_Ethe	2	○	●	SPI	Connect to RESET pin on both LoRa and Ethernet module
DI0_LoRa	34	●	○	Inerrupt	Connect to DI0 pin on LoRa module
ESPrtk IO Name	**GPIO**	**Input Pullup**	**Out**	**Type**	**Function**
RX_UART0	3	●	○	UART	Flash/Update Firmware .- Command control – IMU Output Connect to TX_UART pin on HOST
TX_UART0	1	○	●	UART	Flash/Update Firmware .- Command control – IMU Output Connect to RX_UART pin on HOST
RX_UART1	17	●	○	UART	Connect to TX_A_UART GNSS RTK Receiver
TX_UART1	5	○	●	UART	Connect to RX_A_UART GNSS RTK Receiver

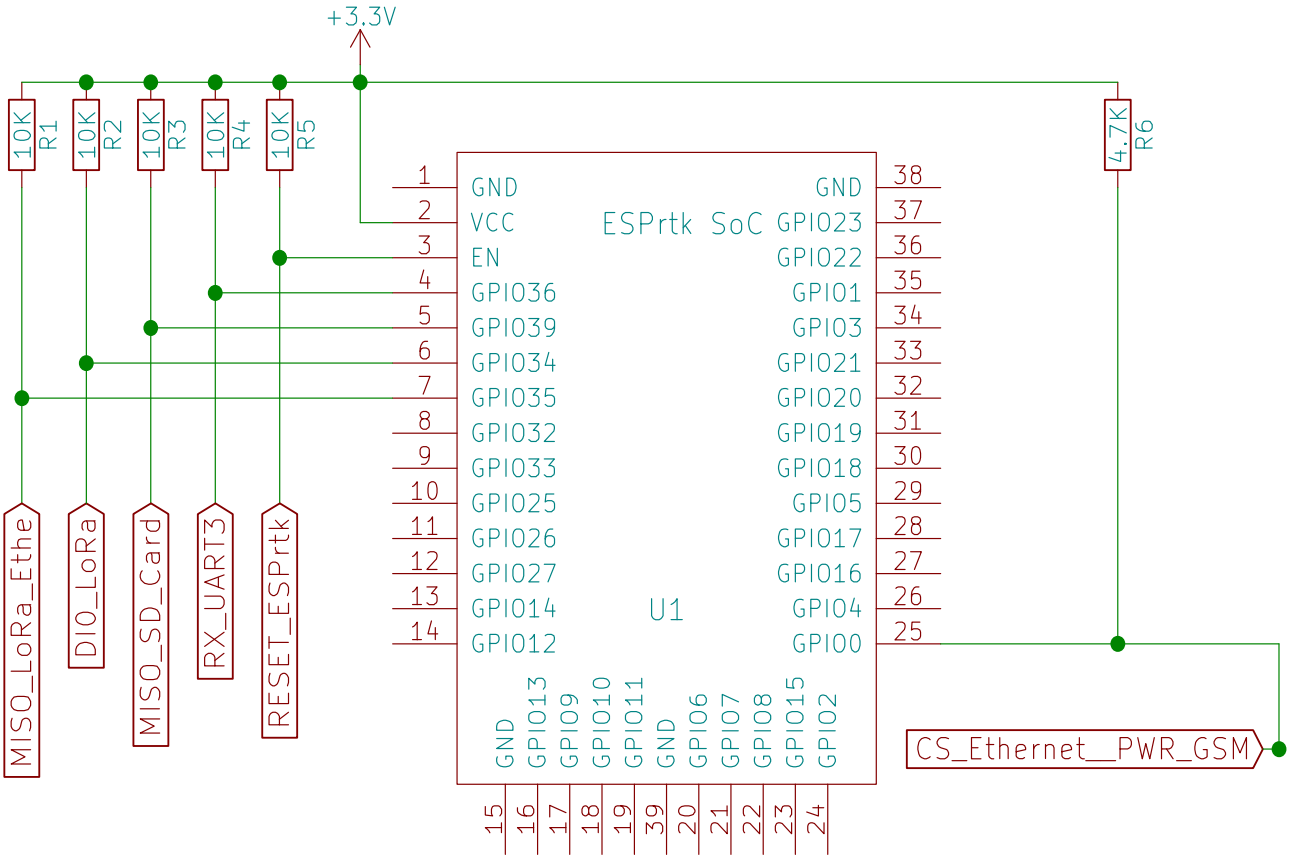
RX_UART2	16	●	○	UART	Connect to TX_B_UART GNSS RTK Receiver
TX_UART2	4	○	●	UART	Connect to RX_B_UART GNSS RTK Receiver
SCL_I2C	22	○	●	I²C / I2S	Connect to SCL pin on Other module (IMU / Sensor / Clock ..) or Connect to NeoPixel Trips 4 LED
SDA_I2C	21	●	●	I²C	Connect to SDA pin on Other module (IMU / Sensor / Clock ..)
RX_UART3	36	●	○	UART	Connect to TX-UART GSM/Cellular 2G/3G/4G/LTE module
ESPrtk IO Name	**GPIO**	**Input Pullup**	**Out**	**Type**	**Function**
MOSI_SD_Card	23	○	●	SPI	Connect to MOSI pin on SD /Micro SD Card module
MISO_SD_Card	39	●	○	SPI	Connect to MISO pin on SD /Micro SD Card module
SCK_SD_Card	18	○	●	SPI	Connect to SCK pin on SD /Micro SD Card module
CONNECT_LED	15	○	●	PWM	Connect to single LED (Pull Down) to show Connection Status
ERROR_LED	12	○	●	PWM	Connect to single LED (Pull Down) to show Error Status
TX_UART3	14	○	●	UART	Connect to RX-UART GSM/Cellular 2G/3G/4G/LTE module
SCL_OLED	27	○	●	I²C	Connect to SCL Pin on OLED SSD1306 /SH1106 Display
SDA_OLED	26	●	●	I²C	Connect to SDA Pin on OLED SSD1306 /SH1106 Display

ESPrtk Pin Out

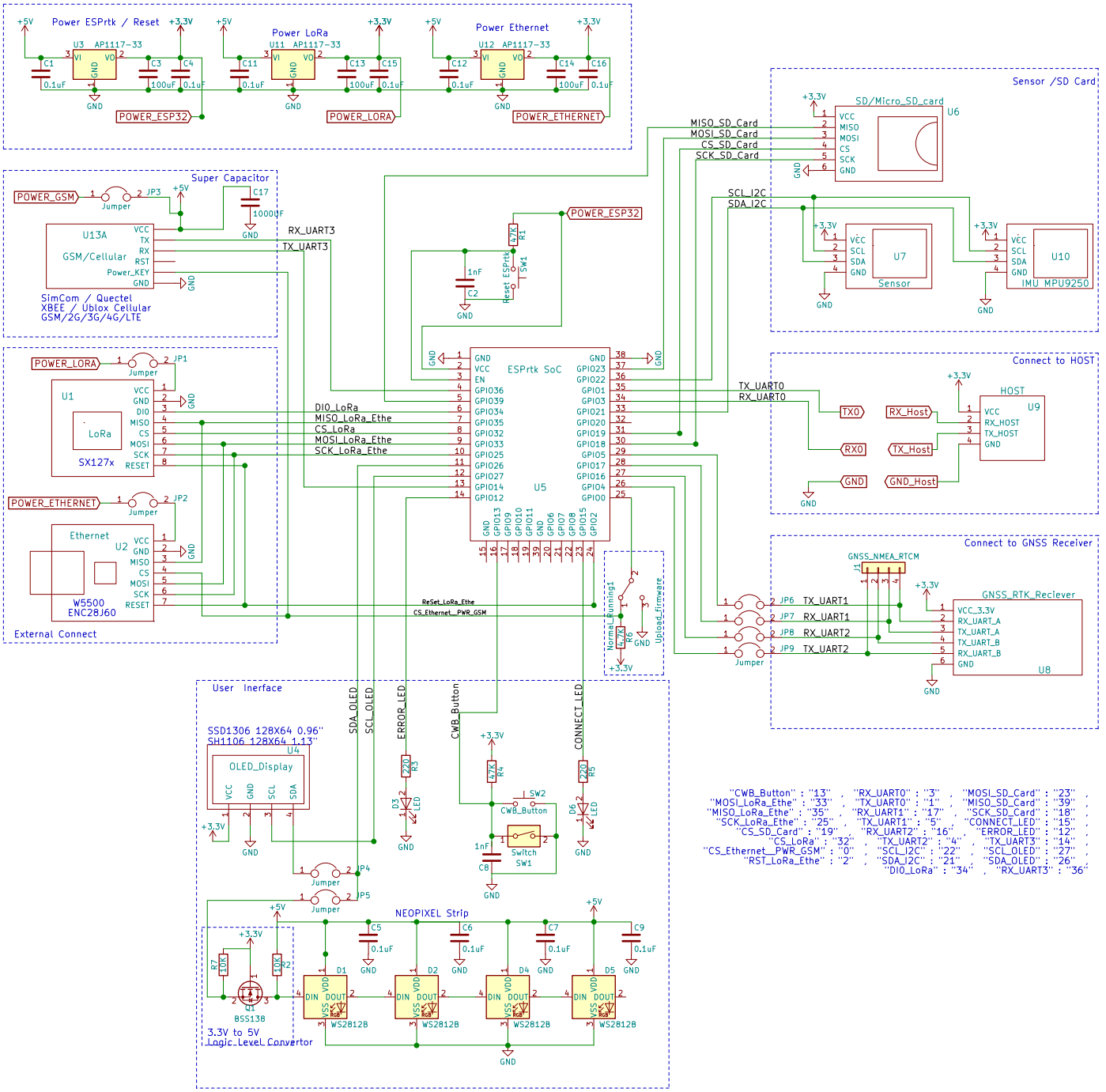


Resistor Pull-Up Require.

The pins RESET, GPIO0, GPIO34, GPIO35, GPIO36, GPIO39 need to connect (always) with a pull-up resistor of 10K ohm (to 4.7K ohm).



Suggest ESPrtk block diagram connection for mutil devices




```
"CWB_Button" : "13" , "RX_UART0" : "3" , "MOSI_SD_Card" : "23" ,
"MOSI_LoRa_Ethe" : "33" , "TX_UART0" : "1" , "MISO_SD_Card" : "39" ,
"MISO_LoRa_Ethe" : "35" , "RX_UART1" : "17" , "SCK_SD_Card" : "18" ,
"SCK_LoRa_Ethe" : "25" , "TX_UART1" : "5" , "CONNECT_LED" : "15" ,
"CS_SD_Card" : "19" , "RX_UART2" : "16" , "ERROR_LED" : "12" ,
"CS_LoRa" : "32" , "TX_UART2" : "4" , "TX_UART3" : "14" ,
"CS_Ethernet_PWR_GSM" : "0" , "SCL_I2C" : "22" , "SCL_OLED" : "27" ,
"RST_LoRa_Ethe" : "2" , "SDA_I2C" : "21" , "SDA_OLED" : "26" ,
"DIO_LoRa" : "34" , "RX_UART3" : "36"
```

Custom Pin Map

This is an advanced feature that allows users to interfere into the output controller by pin mapping on ESP32 to ESPrtk ..

In this way, the selection and placement of the output pins on ESP32 will not depend on the default configuration, making it easier for users to redesign their own ESPrtk PCB board.

Besides helping make PCB design more flexible, this feature allows ESPrtk to be compatible on all hardware boards with different pinout designs.

Users can change the pinmap configuration of ESPrtk with UART Configure or WEB Configure.

Configure	JSON Configure

Default from factory	{
	"CWB_Button" : "13" , "RX_UART0" : "3" , "MOSI_SD_Card" : "23" ,
	"MOSI_LoRa_Ethe" : "33" , "TX_UART0" : "1" , "MISO_SD_Card" : "39" ,
	"MISO_LoRa_Ethe" : "35" , "RX_UART1" : "17" , "SCK_SD_Card" : "18" ,
	"SCK_LoRa_Ethe" : "25" , "TX_UART1" : "5" , "CONNECT_LED" : "15" ,
	"CS_SD_Card" : "19" , "RX_UART2" : "16" , "ERROR_LED" : "12" ,
	"CS_LoRa" : "32" , "TX_UART2" : "4" , "TX_UART3" : "14" ,
	"CS_Ethernet__PWR_GSM" : "0" , "SCL_I2C" : "22" "SCL_OLED" : "27" ,
	"RST_LoRa_Ethe" : "2" , "SDA_I2C" : "21" , "SDA_OLED" : "26" ,
	"DI0_LoRa" : "34" , "RX_UART3" : "36"
	}

Configure	**JSON Configure**

Example configure of User	{
	"CWB_Button" : "36" , "RX_UART0" : "3" , "MOSI_SD_Card" : "15" ,
	"MOSI_LoRa_Ethe" : "27" , "TX_UART0" : "1" , "MISO_SD_Card" : "2" ,
	"MISO_LoRa_Ethe" : "19" , "RX_UART1" : "34" , "SCK_SD_Card" : "14" ,
	"SCK_LoRa_Ethe" : "5" , "TX_UART1" : "4" , "CONNECT_LED" : "17" ,
	"CS_SD_Card" : "13" , "RX_UART2" : "35" , "ERROR_LED" : "16" ,
	"CS_LoRa" : "18" , "TX_UART2" : "12" , "TX_UART3" : "25" ,
	"CS_Ethernet__PWR_GSM" : "0" , "SCL_I2C" : "33" , "SCL_OLED" : "22" ,
	"RST_LoRa_Ethe" : "23" , "SDA_I2C" : "32" , "SDA_OLED" : "21" ,
	"DI0_LoRa" : "26" , "RX_UART3" : "39"
	}

Exceptions .

ESPrtk supports changing all output pins. However, not all configurations are accepted.

-Some GPio pins of ESP32 cannot be used for ESPrtk if it does not fully support. See table below

GPIO (ESP32)	**…map to ESPrtk (Input)**	**.. map to ESPrtk (Output)**
Input-Output	Yes	Yes
Input Only	Yes	No
Output Only	No	Yes

Sometime the configuration is successfully saved (passes the Logic test) but ESPrtk will fail or not work properly, this depends on how the user configures and how ESP32 work with that configuration.

So users need to know the limitations of output pin on ESP32. Below is a brief summary.

GPIO	Input	Output	Notes
0	●	●	**Pulled up when start**
1	●	●	**Debug output at boot . Update firmware to ESPrtk.**
2	●	●	
3	●	●	**Debug output at boot Update firmware to ESPrtk. HIGH at boot**
4	●	●	
5	●	●	
6	○	○	**Connected to the integrated SPI flash**
7	○	○	**Connected to the integrated SPI flash**
8	○	○	**Connected to the integrated SPI flash**
9	○	○	**Connected to the integrated SPI flash**
10	○	○	**Connected to the integrated SPI flash**
11	○	○	**Connected to the integrated SPI flash**
12	●	●	**Boot fail if pulled high**
13	●	●	
14	●	●	
15	●	●	

GPIO	Input	Output	Notes
16	●	●	
17	●	●	
18	●	●	
19	●	●	
21	●	●	
22	●	●	
23	●	●	
25	●	●	
26	●	●	
27	●	●	
32	●	●	
33	●	●	
34	●	○	input only
35	●	○	input only
36	●	○	input only
39	●	○	input only

ESPrtk IO Name	0	1	2	3	4	5	12	13	14	15	16	17	18	19	21	22	23	25	26	27	32	33	34	35	36	39
CWB_Button (IN)	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
MOSI_LoRa_Ethe	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
MISO_LoRa_Ethe (IN)	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
SCK_LoRa_Ethe	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
CS_SD_Card	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
CS_LoRa	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
CS_Ethernet__PWR_GSM	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○

RST_LoRa_Ethe	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
DI0_LoRa (IN)	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
RX_UART0 (IN)	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
TX_UART0	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
RX_UART1 (IN)	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
TX_UART1	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
RX_UART2 (IN)	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
TX_UART2	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
SCL_I2C	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
SDA_I2C	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
RX_UART3 (IN)	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
MOSI_SD_Card	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
MISO_SD_Card (IN)	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
SCK_SD_Card	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
CONNECT_LED	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
ESPrtk IO Name	**0**	**1**	**2**	**3**	**4**	**5**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**21**	**22**	**23**	**25**	**26**	**27**	**32**	**33**	**34**	**35**	**36**	**39**
ERROR_LED	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
TX_UART3	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
SCL_OLED	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○
SDA_OLED	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○	○

UART / SD Card Driver on ESPrtk .

Besides the ability to map input / output pins.

For UART connection, there is an additional secondary mapping function, UART-Soft-Switch, for faster communication switching.

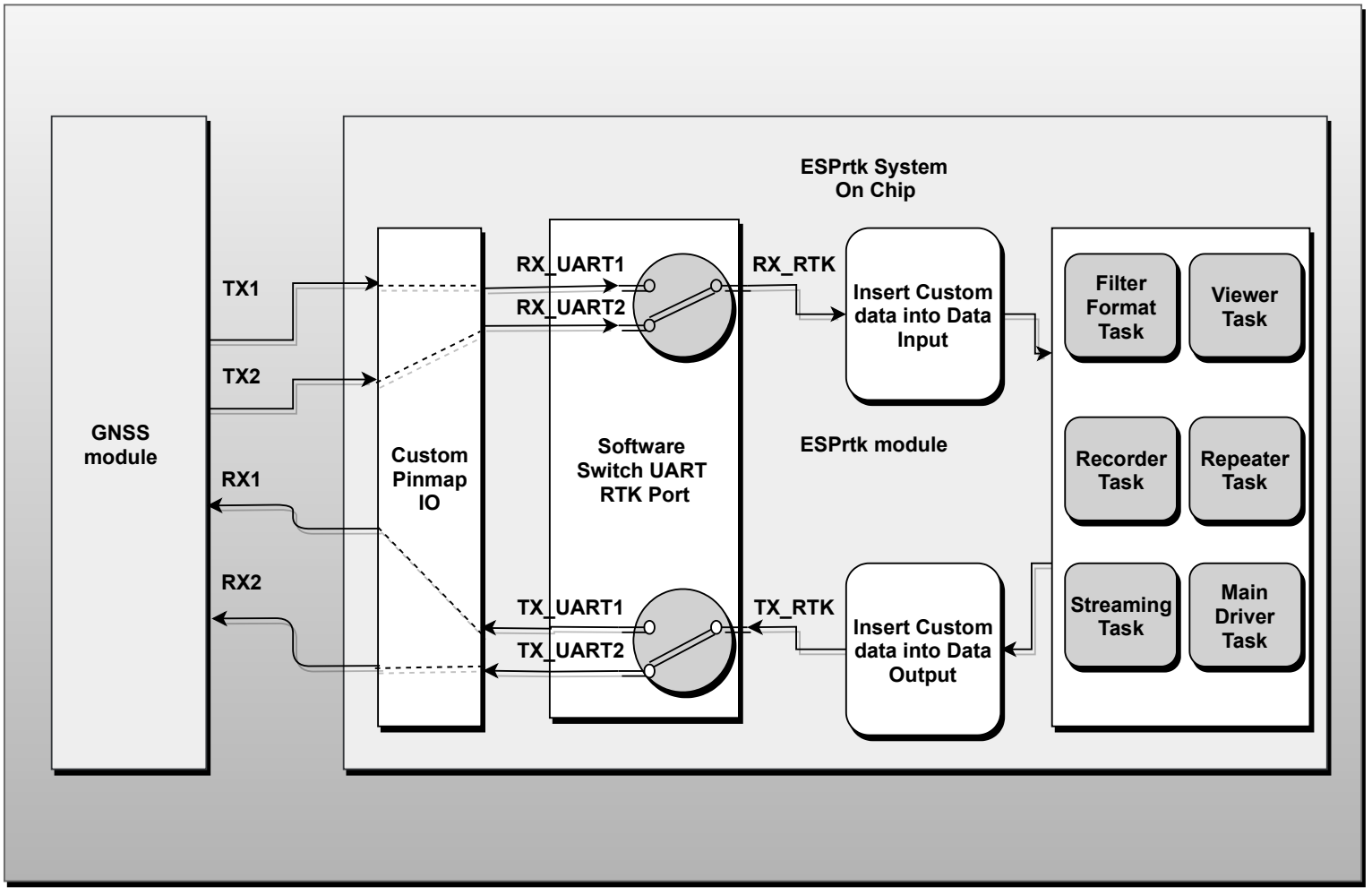
When running the application, the UART port with actual names are TX_UART_RTK and RX_UART_RTK will map to port TX_UART1 (or TX_UART2) and to port RX_UART1 (or RX_UART2).

For Base / Rover RTK applications, UART Driver also supports a lot of input data filters, supports inserting any data (default data) of users up to 500 Bytes on both TX / RX.The insert processing will be done automatically to ensure the main data stream will not be interrupted.

Data from filters and inserts will be forwarded to other processing tasks, including the Recorder Task.

Real-time capture and data storage will be performed by SD_Card Driver. Here, NMEA / RTCM / RAW,... data continues to be processed before storing.

SD_Card Driver on ESPrtk 3.0.0 supports many different recording types, file types, automation and card data protection.



Display for user communication and Sensor.

When running Main Action mode, information about the status of ESPrtk such as Error, Thread Status, IMU Sensor Data, PING response, Control Command response, etc. will be exported to TX_UART0 port as ASCII character. (can read and understand).

All outputs are assigned an ID to distinguish and in the proprietary format of ESPrtk TM.

In addition, information exported on the TX_UART0 port can also be controlled (turned on / off) by default settings or turned on / off at run time via the Control Command from HOST.

For intuitive status display, ESPrtk supports display by 128x64 OLED display, NEOPIXEL LED strip and status LED.

ESPrtk also supports IMU peripheral sensor. (Accelerometer / Gyroscope / Magnetometer). High-precision IMU / INS data is processed in real time with low delay, sensor read-up frequency can up to 200Hz.

The CWB_Button use to start WEB Configure mode by hold it in 3 second (set to Low level).

Symbol Type Value

Type value	Characters contain	Example	Notes
[0 → 9]	0123456789	0 , 123, 457	Number only
[a → z]	abcdefghijklmnopqrstuvwxyz	esprtk	
[A → Z]	ABCDEFGHIJKLMNOPQRSTUVWXYZ	ESPRTK	
[INTEGER]	-0123456789	0 , -123, -457	Number only and '-'
[FLOAT]	-0123456789.	0.123 , -12.23 , 67.97	Number only and '-' and '.'
[BIT]	01	011101, 1111001, 001	'0' and '1' only
[HEX_Up]	0123456789ABCDEF	A0,B0, C045, AD7C, A8E4C0D2	1 byte of data must be represented by 2 characters.
[HEX]	0123456789acbdefABCDEF	A0,b7, a3ee, B3f5, A8E4c0d2	1 byte of data must be represented by 2 characters.
[<CR><LF>]	2 byte value are 0x0D , 0x0A		
[PRINT]	abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 "#%&'()+,-./:;<=>@^_`{}~	Hello_ESPrtk_01 , Hello-ESPrtk-02 , Hello ESPrtk $# ,	Not allow contain <CR><LF>, $*|
[STRING]	AbcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 -_+(.):,	Hello_ESPrtk_01 , Hello-ESPrtk-02 ,	Not allow contain <CR><LF>, **WhiteSpace , !=&"~<\/>?|%#[]{}^$***
[PASS_A]	**[STRING]**		Not allow contain <CR><LF>, **, !=&"~<\/>?|%#[]{}^$***
[PASS_B]	**[PASS_A]**		Not allow contain WhiteSpace
[PASS_C]	**[PASS_A]**		Must contain atleast 1 Upper Case and 1 Lower Case and 1 Number Min length 8 chars
[PASS_D]	**[PASS_C]**		Not allow contain WhiteSpace
[AT_Code]	**[PRINT]**	[AT+CMT=0]	Not allow contain **!&~?|%**

Format the message frame communicating on the port UART0.

Message Frame:

<Header> + <Length_Payload> + <Payload> + <*CRC> + <CR> <LF>

Example : **$ESPLOG|25|2|INFO|GO to MAIN ACTION|*6360**

Label	Characters allowed to contain.
<Header>	**[0 → 9] [a → z] [A → Z]**
<Length_Payload>	**[0 → 9]**
<Payload>	**[ASC]**
<*CRC>	**[HEX_Up]**
<CR><LF>	**[<CR><LF>]**

Label	Sub	Min Bytes	Max Bytes	Describe
<Header>	'$' + 'Header_Name' + '|'	3	10	Start Message
<Length_Payload>	'Length_Payload' +'|'	2	10	Numbers Bytes of Payload
<Payload>	'Payload'	1	Length_Payload	Payload data
<*CRC>	'*' + 'CRC_16'	5	5	Checksum of Message
<CR><LF>	0x0D + 0x0A	2	2	End Message

	<Header>	<Length_Payload>	<Payload>	<*CRC>	<CR><LF>
Ex 1	$ESPLOG|	25|	2|INFO|GO to MAIN ACTION|	*6360	\r\n
Ex 2	$ESPLOG|	34|	1|ERROR|0,0,G-G-G-G, No ERROR|	*7A58	\r\n

Ex 3	$ESP_OK|	5|	X|_||	*5309	\r\n
Ex 4	$ESP_OK|	14|	W|m|3|0|0|0|0|	*6D76	\r\n

Length_Payload calculate.

- Example message : **$ESP_OK|14|W|m|3|0|0|0|0|*6D76**
- Then Payload range is : **W|m|3|0|0|0|0|** **and** Total of characters (Length) is **14**

CRC calculate.

CRC_Range calculate from after '$' to begin '*' (not contain '$' and '*')

'Header_Name' + '|'+ <Length_Payload> + <Payload>

- Example message : **$ESP_OK|14|W|m|3|0|0|0|0|*6D76**
- Then CRC Range is: **ESP_OK|14|W|m|3|0|0|0|0|**

CRC16 Algorithm calculate in C/C++ for ESPrtk frame:

```
const char  MSG[]= "$ESP_OK|14|W|m|3|0|0|0|0|*????\r\n";

// Calculate CRC
uint8_t FIRST_CRC  = 0 ;
uint8_t SECOND_CRC = 0 ;
uint16_t  i = 0 ;

while ( MSG [i]  !=  '*' ){
  // CRC_Range calculate from after '$' to begin '*'  ( not contain '$' and '*')
  if (  MSG [i]  != '$' ){
    FIRST_CRC    ^=  ( uint8_t  ) MSG [i];
    SECOND_CRC ^=  ( uint8_t ) FIRST_CRC;
  }
  i++;
}

//Format CRC to [HEX_Up] string and store to buffer
char CRC16_OUT  [4+1] ;
sprintf (  CRC16_OUT   ,  "%02X%02X" , FIRST_CRC , SECOND_CRC );

//Print out result
printf ( "\n Your CRC16 is %s \n", CRC16_OUT);
```

```
//Result should be [ Your CRC16 is 6D76 ]
```

ESPrtk support suggestions for fixing errors.

When ESPrtk receives an incorrectly structured message frame, it returns a NACK message.

If the message is valid and corrupted, ESPrtk will give an error and suggest correction.

A valid structured message (minimum) must include **<Header> <Length_Payload> <Payload> <* CRC>**, for example: **$ESP_OK|??|?|?|*????**

Note : The error correction suggestion function is only fully supported when running in UART Configure mode.

Examples:

HOST Send	$ESP_OK|??|?|?|*????
Respond	NACK :[?] LENGTH_MESSAGE invalid (not a number) - [?] Do you mean LENGTH_MESSAGE = 4

HOST Send	$ESP_OK|4|?|?|*????
Respond	NACK : Your_CRC = [????]: [?] Do you mean Checksum_CRC = 295A

HOST Send	$ESP_OK|?|P|_||*????
Respond	NACK :[?] LENGTH_MESSAGE invalid (not a number) - [?] Do you mean LENGTH_MESSAGE = 5

HOST Send	$ESP_OK|5|P|_||*????
Respond	NACK : Your_CRC = [????]: [?] Do you mean Checksum_CRC = 5B01

HOST Send	$ESP_OK|5|P|_||*5B01

Respond	$ESP_OK|5|P|_||*5B01

Types of message frame

Type frame	<Header>	Describe
LOG_FRAME	$ESPLOG|	Output status, information of ESPrtk
IMU_FRAME	$ESPIM1|	IMU / INS data output type 1
	$ESPIM2|	IMU / INS data output type 2
COMMAND_FRAME	$ESP_OK|	Respond to HOST with ACK status (non encrypt Payload) . *
	$ESS_OK|	Respond to HOST with ACK status (BitWise encrypt Payload).*
	$AES_OK|	Respond to HOST with ACK status (AES256 encrypt Payload).*
	$ESP_ER|	Respond to HOST with NACK status (non encrypt Payload) .*
	$ESS_ER|	Respond to HOST with NACK status (BitWise encrypt Payload).*
	$AES_ER|	Respond to HOST with NACK status (AES256 encrypt Payload).*

** Tip :*
ACK status : ESPrtk accept command message from HOST.
NACK status: ESPrtk receive a wrong format command message from HOST, so ESPrtk will ignore this command (not accept).
BitWise/AES256 encrypt payload : Payload data will be encrypt before export , please read more to learn about this feature.

$ESPLOG Message

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESPLOG|xx|	Type Log in Number	Type Log in String	Infor 1	Infor 2	Infor 3	Infor 4	...

PL0	Describe	Example
0	ESPrtk System	$ESPLOG|xx|0|SYSTEM|...
1	Error status	$ESPLOG|xx|1|ERROR|...
2	Text , Log, Information...	$ESPLOG|xx|2|INFO|...
3	Go to function status	$ESPLOG|xx|3|GOIN|...
4	Wifi Scan Result	$ESPLOG|xx|4|WFSCAN|..
5	Broker MQTT Status	$ESPLOG|xx|5| BROKER|...
6	Before print NMEA from Rover	$ESPLOG|xx|6| RVNMEA|...
7	Configure information detail	$ESPLOG|xx|7|CONFIG|...

$ESPIM1 Message

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL0	[FLOAT]			0.00	0	15	Raw Gyroscope X Axis	deg/s
PL1	[FLOAT]			0.00	0	15	Raw Gyroscope Y Axis	deg/s
PL2	[FLOAT]			0.00	0	15	Raw Gyroscope Z Axis	deg/s
PL3	[FLOAT]			0.00	0	15	Raw Magnetometer X Axis (micro-Tesla)	uT
PL4	[FLOAT]			0.00	0	15	Raw Magnetometer Y Axis (micro-Tesla)	uT
PL5	[FLOAT]			0.00	0	15	Raw Magnetometer Z Axis (micro-Tesla)	uT
PL6	[FLOAT]			0.00	0	15	Raw Accelerometer X Axis (scale 0.1 m/s^2)	-
PL7	[FLOAT]			0.00	0	15	Raw Accelerometer Y Axis (scale 0.1 m/s^2)	-
PL8	[FLOAT]			0.00	0	15	Raw Accelerometer Z Axis (scale 0.1 m/s^2)	-
PL9	[FLOAT]			0.00	0	15	Temperature sensor	°C
PL10	[FLOAT]			0.00	0	15	Magnetometer X Axis + Calibrated	-
PL11	[FLOAT]			0.00	0	15	Magnetometer Y Axis + Calibrated	-
PL12	[FLOAT]			0.00	0	15	Magnetometer Z Axis + Calibrated	-

Example:

```
$ESPIM2|81|0.16|-0.09|-0.98|0.04|168.72|-172.96|-18.37|39.75|||||166.81|-163.44|19.96|53.70|*5172
$ESPIM1|81|82.32|199.63|172.13|-2.40|-7.20|-50.41|0.54|-0.52|-0.57|29.17|-2.40|-7.20|-50.41|*186D
$ESPIM2|82|0.35|-0.25|-0.88|0.20|136.90|-141.23|-31.31|21.44|||||139.56|-141.49|-30.51|22.45|*750C
$ESPIM1|84|178.90|116.04|202.62|17.55|-21.91|-30.16|0.57|-0.76|-0.12|29.16|17.55|-21.91|-30.16|
*2071
$ESPIM2|82|0.57|-0.51|-0.57|0.30|79.64|-100.98|-20.24|334.15|||||83.66|-106.79|-20.59|315.65|*7D05
$ESPIM1|82|117.38|127.38|185.91|37.51|-10.50|-8.85|-0.09|-0.79|0.14|29.16|37.51|-10.50|-8.85|*2211
$ESPIM2|77|0.66|-0.60|-0.20|0.40|51.81|-78.48|12.85|316.72|||||52.77|-79.97|6.36|316.71|*4774
$ESPIM1|76|-12.68|51.89|19.82|39.61|10.80|2.55|-0.66|-0.60|0.13|29.16|39.61|10.80|2.55|*3F7F
$ESPIM2|77|0.63|-0.64|0.03|0.45|41.11|-76.36|38.07|295.72|||||41.76|-76.70|35.26|303.99|*5E7E
$ESPIM1|79|-170.67|-256.16|-166.71|41.41|8.25|0.60|-0.30|-0.80|0.21|29.16|41.41|8.25|0.60|*004B
$ESPIM2|78|0.59|-0.60|-0.19|0.50|62.77|-77.32|21.63|314.30|||||54.44|-77.11|23.91|309.77|*7C09
$ESPIM1|86|-160.12|-91.22|-182.01|30.01|-17.70|-25.81|0.61|-0.86|-0.59|29.17|30.01|-17.70|-25.81|
*3C50
$ESPIM2|81|0.45|-0.47|-0.71|0.25|99.49|-120.54|-23.98|344.87|||||93.43|-97.15|-19.13|336.54|*433B
```

$ESPIM2 Message

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL0	[FLOAT]	-1.00	1.00	0.00	0	15	DMP-Quaternion - QW	-
PL1	[FLOAT]	-1.00	1.00	0.00	0	15	DMP-Quaternion - QX	-
PL2	[FLOAT]	-1.00	1.00	0.00	0	15	DMP-Quaternion - QY	-
PL3	[FLOAT]	-1.00	1.00	0.00	0	15	DMP-Quaternion - QZ	-
PL4	[FLOAT]	0.00	360.0	0.00	0	15	DMP-Euler YAW	deg
PL5	[FLOAT]	-180.0	180.0	0.00	0	15	DMP-Euler PITCH	deg
PL6	[FLOAT]	-90.0	90.0	0.00	0	15	DMP-Euler ROLL	deg
PL7	[FLOAT]	0.00	360.0	0.00	0	15	Compass Heading (using magnetometer calibrated) + Tilt compensation	deg
PL8	[FLOAT]	-1.00	1.00	0.00	0	15	DMP-Quaternion -QW+ Extended Kalman Filter	-
PL9	[FLOAT]	-1.00	1.00	0.00	0	15	DMP-Quaternion -QX+ Extended Kalman Filter	-
PL10	[FLOAT]	-1.00	1.00	0.00	0	15	DMP-Quaternion -QY+ Extended Kalman Filter	-
PL11	[FLOAT]	-1.00	1.00	0.00	0	15	DMP-Quaternion -QZ+ Extended Kalman Filter	-
PL12	[FLOAT]	0.00	360.0	0.00	0	15	DMP-Euler YAW + Extended Kalman Filter	deg
PL13	[FLOAT]	-180.0	180.0	0.00	0	15	DMP-Euler PITCH + Extended Kalman Filter	deg
PL14	[FLOAT]	-90.0	90.0	0.00	0	15	DMP-Euler ROLL + Extended Kalman Filter	deg
PL15	[FLOAT]	0.00	360.0	0.00	0	15	Compass Heading (using magnetometer calibrated) + Tilt compensation +Extended Kalman Filter	deg

Example:

```
$ESPIM2|81|0.16|-0.09|-0.98|0.04|168.72|-172.96|-18.37|39.75|||||166.81|-163.44|19.96|53.70|*5172
$ESPIM2|82|0.35|-0.25|-0.88|0.20|136.90|-141.23|-31.31|21.44|||||139.56|-141.49|-30.51|22.45|*750C
$ESPIM2|82|0.57|-0.51|-0.57|0.30|79.64|-100.98|-20.24|334.15|||||83.66|-106.79|-20.59|315.65|*7D05
$ESPIM2|77|0.66|-0.60|-0.20|0.40|51.81|-78.48|12.85|316.72|||||52.77|-79.97|6.36|316.71|*4774
$ESPIM2|77|0.63|-0.64|0.03|0.45|41.11|-76.36|38.07|295.72|||||41.76|-76.70|35.26|303.99|*5E7E
$ESPIM2|78|0.59|-0.60|-0.19|0.50|62.77|-77.32|21.63|314.30|||||54.44|-77.11|23.91|309.77|*7C09
```

$ESP_xx - $ESS_xx - $AES_xx Message

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	Type Command	ID Command	Infor 1	Infor 2	Infor 3	Infor 4	...

PL0	Describe Type Command	Command
'N'	Start WEB Configure mode	$ESP_OK|5|N|_||*451F
'O'	Start UART Configure mode	$ESP_OK|5|O|_||*441E
'X'	Start Main Action mode	$ESP_OK|5|X|_||*5309
'P'	PING to ESPrtk	$ESP_OK|5|P|_||*5B01
'Q'	Quit UART Configure mode	$ESP_OK|5|Q|_||*5A00
'R'	Read data configure on ESPrtk	$ESP_OK|xx|R|….
'W'	Write configure to ESPrtk	$ESP_OK|xx|W|….
'E'	Export configure as example	$ESP_OK|5|E|_||*4E14
'T'	Hardware control	$ESP_OK|xx|T|….

PL0	Type Command	UART Configure Mode support this command ?	Main Action Mode support this command ?	WEB Configure Mode support this command ?	WEB Configure Mode Support ?- Command Tester Tool
'N'	Start WEB Configure mode	●	●	●	●
'O'	Start UART Configure mode	●	●	●	●
'X'	Start Main Action mode	●	●	●	●
'P'	PING to ESPrtk	●	●	●	●
'Q'	Quit UART Configure mode and jump to Main Action mode	●	○	○	○
'R'	Read data configure on ESPrtk	●	○	○	●
'W'	Write configure to ESPrtk	●	○	○	●
'E'	Export configure as example	●	○	○	●
'T'	Hardware control	●	○	○	○
Exception support					
'T'	Hardware control (Control type PRINT output on TX_UART0)	●	●	●	●
'T'	Hardware control (Control GPIO ESPrtk)	●	●	●	●
'R'	Request ERROR_SYSTEM	●	●	●	●

PL0	HOST send command Start WEB Configure mode	ESPrtk respond		
		ACK	NACK	Max delay
'N'	$ESP_OK|5|N|_||*451F	$ESP_OK|5|N|_||*451F	$ESP_ER|xx|N|..	8 Second

PL0	HOST send command Start UART Configure mode	ESPrtk respond		
		ACK	NACK	Max delay
'O'	$ESP_OK|5|O|_||*441E	$ESP_OK|5|O|_||*441E	$ESP_ER|xx|O|..	4 Second

PL0	HOST send command Start Main Action mode	ESPrtk respond		
		ACK	NACK	Max delay
'X'	$ESP_OK|5|X|_||*5309	$ESP_OK|5|X|_||*5309	$ESP_ER|xx|X|..	4 Second

PL0	HOST send command PING to ESPrtk	ESPrtk respond		
		ACK	NACK	Max delay
'P'	$ESP_OK|5|P|_||*5B01	$ESP_OK|5|P|_||*5B01	$ESP_ER|xx|P|..	2 Second

PL0	HOST send command Quit UART Configure mode	ESPrtk respond		
		ACK	NACK	Max delay
'Q'	$ESP_OK|5|Q|_||*5A00	$ESP_OK|5|Q|_||*5A00	$ESP_ER|xx|Q|..	1 Second

Start Main Action mode.

By default, ESPrtk after supplying power (or RESET) will enter Main Action mode.

Use ESPrtk command to start Main Action mode

Use ESPrtk command control	
Step 0	ESPrtk after supplying power.
Step 2	HOST send command : **$ESP_OK|5|X|_||*5309**
Done	ESPrtk will start go to Main Action mode.
Success	The CONNECT_LED blink slow. ESPrtk respond on TX_UART0: **$ESP_OK|5|X|_||*5309**

Use the RESET button to start Main Action mode

Method 2: Use only the CWB button	
Step 0	ESPrtk after supplying power.
Done	Press RESET button (set EN pin to Low level) , then release. → ESPrtk will start go to Main Action mode.
Success	The CONNECT_LED will light continuously . ESPrtk respond on TX_UART0: **$ESP_OK|5|X|_||*5309**

Start UART Configure mode for Write configure data.

Use ESPrtk command to start UART configure

Use ESPrtk command control	
Step 0	ESPrtk after supplying power.
Step 2	HOST send command : **$ESP_OK|5|O|_||*441E**
Done	ESPrtk will start go to WEB Configure mode. The CONNECT_LED blink fast.
Success	The CONNECT_LED will light continuously . ESPrtk respond on TX_UART0: **$ESP_OK|5|O|_||*441E**

Command Write Configure ESPrtk

Tip : To use these command , ESPrtk need running at UART Configure mode !

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	ID Command	Infor 1	Infor 2	Infor 3	Infor 4	...

PL0	PL1	Describe Type Command	Example
'W'	'c'	Write YOUR_PROFILE 's configure data	
'W'	'd'	Write ACTION_PLANNING 's configure data	
'W'	'e'	Write INTNERNET 's configure data	
'W'	'f'	Write RADIO_RTK 's configure data	
'W'	'g'	Write MQTT_RTK 's configure data	
'W'	'h'	Write NTRIP_CLIENT 's configure data	
'W'	'i'	Write NTRIP_MASTER 's configure data	
'W'	'j'	Write UART_CONFIGURE 's configure data	
'W'	'k'	Write BLUETOOTH 's configure data	
'W'	'l'	Write DISPLAY_VIEWER 's configure data	

‘W’	‘m’	Write EVENT_LOG ‘s configure data	
‘W’	‘n’	Write SIMPLE_DEBUG ‘s configure data	
‘W’	‘o’	Write SD_CARD ‘s configure data	
‘W’	‘p’	Write IMU_MPU9250 ‘s configure data	
‘W’	‘q’	Write GPIO_PIN_MAP ‘s configure data	
‘W’	‘r’	Write SYSTEM_Other_Mode ‘s configure data	
‘W’	‘s’	Write GSM_CELLULAR ‘s configure data	
‘W’	‘t’	Write STREAM_MANAGER‘s configure data	
‘W’	‘u’	Write POSITION_ANTENNA‘s configure data	
‘W’	‘v’	Write WIFI_TCP_UDP‘s configure data	
‘W’	‘w’	Write NTRIPCASTER_Admin‘s configure data	
‘W’	‘x’	Write NTRIPCASTER_ACC‘s configure data	

Command Write Configure - YOUR_PROFILE

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'c'	Infor 1	Infor 2	Infor 3	Infor 4	...

Note: This configure only work on WEB Configure interface !

Name CFG : YOUR_PROFILE
Number of field values : 9

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	PRINT			ESPrtk : [SHORT_ID]	3	30	Name Hotspot	
PL3	PASS_D			def123DEF	8	40	Password Hotspot	
PL4	INTEGER	0	1	0			Enable Hotspot Security	
PL5	PASS_B			admin	1	40	Name Login Profile	
PL6	PASS_D			abc123ABC	8	40	Password Profile	
PL7	PRINT			Hello member	0	15	Nickname	
PL8	PRINT			ESPrtk new user	0	40	Introduce	
PL9	INTEGER	0	1	1	-	-	Allow create new Profile	
PL10	INTEGER	0	4	0	-	-	Type CSS theme WEB Configure's interface	
PL11								
PL12								
PL13								
PL14								
PL15								
PL16								

PL0	PL1	PL2	Enable Hotspot Security
'W'	'c'	'0'	Disable Hotspot Security
		'1'	Enable Hotspot Security

PL0	PL1	PL9	Allow create new Profile
'W'	'c'	'0'	Not allow create new Profile
		'1'	Allow create new Profile

PL0	PL1	PL10	Type CSS theme WEB Configure's interface
'W'	'c'	'0'	Default – BGW theme
		'1'	BWR theme
		'2'	BGG theme
		'3'	Other theme
		'4'	Other theme

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'c'	STRING	-	-	1	1	ID Configure : YOUR_PROFILE		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Note: This configure only work on WEB Configure interface !

Message :	
HOST Send	$ESP_OK|80|W|c|ESPrtk : IHGG4|def123DEF|0|admin|abc123ABC|ESPrtk new user|Hello member|1|0|*134C
ACK Respond	$ESP_OK|46|W|c|1|(ID : YOUR_PROFILE) : Saved Success .|*3B1B
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - ACTION_PLANNING

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'd'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : ACTION_PLANNING
Number of field values : 9

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	1	10000	10	-	-	Max Rover	
PL3	INTEGER	1	10000	1	-	-	ID Rover	
PL4	INTEGER	0	14	8	-	-	Type Baudrate UART_0	
PL5	INTEGER	0	14	6	-	-	Type Baudrate UART_1	
PL6	INTEGER	0	14	6	-	-	Type Baudrate UART_2	
PL7	INTEGER	0	1	1	-	-	Map TX-UART-RTK	
PL8	INTEGER	0	1	0	-	-	Map RX-UART-RTK	
PL9	INTEGER	0	17	0	-	-	Main Action	
PL10	BIT	-	-	0000000000000000000	19	19	Other action	

TIP: Note, when Internet method using GSM/Cellular module, the UART2 baudrate must equal to UART1 baudrate.

PL0	PL1	PL4	Type Baudrate UART_0	Unit
		PL5	Type Baudrate UART_1	
		PL6	Type Baudrate UART_2	
'W'	'd'	0	1200	bps
		1	2400	bps
		2	4800	bps
		3	9600	bps
		4	19200	bps
		5	38400	bps
		6	57600	bps
		7	115200	bps
		8	230400	bps
		9	250000	bps
		10	460800	bps
		11	500000	bps
		12	921600	bps
		13	1000000	bps
		14	2000000	bps

TIP: Note, when Internet method using GSM/Cellular module, the UART2 baudrate must equal to UART1 baudrate.

PL0	PL1	PL7	Map TX-UART-RTK	Unit
'W'	'd'	0	Map TX-UART-RTK to TX_UART_1 port	
		1	Map TX-UART-RTK to TX_UART_2 port	

PL0	PL1	PL8	Map RX-UART-RTK	Unit
'W'	'd'	0	Map RX-UART-RTK to RX_UART_1 port	
		1	Map RX-UART-RTK to RX_UART_2 port	

PL0	PL1	PL9	Main Action	Unit
'W'	'd'	0	No Main Action	
		1	MQTT_Base	
		2	NTRIP_Base	
		3	Radio_Base	
		4	MQTT_Rover	
		5	NTRIP_Rover	
		6	Radio_Rover	
		7	UART_to_Bluetooth	
		8	UART_to_RF_Base	
		9	UART_to_RF_Rover	
		10	Bluetooth_to_RF_Base	
		11	Bluetooth_to_RF_Rover	
		12	MQTT_Rover_Repeater	
		13	NTRIP_Rover_Repeater	
		14	Radio_Rover_Repeater	
		15	UART_Base_to_UART_Rover	
		16	NTRIP_CASTER	
		17	UART_to_WIFI_Bridge	

PL0	PL1	PL10	Other action
'W'	'd'	BIT1	Export IMU Data to TX_UART_0 port
		BIT2	Save NMEA/RTCM/RAW/.. Data (from RX-UART-RTK) to SD Card
		BIT3	Send NMEA/RTCM/RAW/.. Data (from RX-UART-RTK) to Bluetooth Client
		BIT4	Base-Export NMEA/GGA data (from Rovers) to TX_UART_0 port
		BIT5	Base-Priority export NMEA Data from Rovers
		BIT6	Rover-Save RTCM/RAW... data (from Server/Radio_Base) to SD Card
		BIT7	Rover-Send GGA message (from RX-UART-RTK) to Server/RTK_Base + Export GGA to TX_UART0
		BIT8	Rover-Send all NMEA messages (from RX-UART-RTK) to Server/RTK_Base + Export NMEA to TX_UART0
		BIT9	Wireless Bridge: Save data send (TX-Wireless-Data) to SD Card
		BIT10	Wireless Bridge: Save data receive (RX-Wireless-Data) to SD Card
		BIT11	Rover Repeat: Send RTCM/RAW... data (from Server/Server/RTK_Base) to Bluetooth Client
		BIT12	Rover Repeat: Send RTCM/RAW... data (from Server/RTK_Base) to RTK_Radio_Rover
		BIT13	Send IMU_1 Data to Bluetooth Client
		BIT14	Send IMU_2 Data to Bluetooth Client
		BIT15	Send Data (from RX-UART-RTK) to WiFi TCP/UDP device
		BIT16	Send IMU_1 Data to WiFi TCP/UDP device
		BIT17	Send IMU_2 Data to WiFi TCP/UDP device
		BIT18	Respond realtime command control on Bluetooth Client
		BIT19	Respond realtime command control on WiFi TCP/UDP

PL0	PL1	PL9	PL10 – BIT Support (1 → 19)																		
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
'W'	'd'	0	●	●	●	○	○	○	●	●	○	○	○	○	●	●	●	●	●	●	●
		1	●	●	●	●	●	○	○	○	○	○	○	○	●	●	●	●	●	●	●
		2	●	●	●	●	●	○	○	○	○	○	○	○	●	●	●	●	●	●	●
		3	●	●	●	●	●	○	○	○	○	○	○	○	●	●	●	●	●	●	●
		4	●	●	●	○	○	●	●	●	○	○	○	○	●	●	●	●	●	●	●
		5	●	●	●	○	○	●	●	●	○	○	○	○	●	●	●	●	●	●	●
		6	●	●	●	○	○	●	●	●	○	○	○	○	●	●	●	●	●	●	●
		7	●	○	○	○	○	○	○	○	●	●	○	○	○	○	●	●	●	●	●
		8	●	○	○	○	○	○	○	○	●	●	○	○	○	○	●	●	●	●	●
		9	●	○	○	○	○	○	○	○	●	●	○	○	○	○	●	●	●	●	●
		10	●	○	○	○	○	○	○	○	●	●	○	○	○	○	●	●	●	●	●
		11	●	○	○	○	○	○	○	○	●	●	○	○	○	○	●	●	●	●	●
		12	●	●	○	●	●	●	●	●	○	○	●	●	●	●	●	●	●	●	●
		13	●	●	○	●	●	●	●	●	○	○	●	●	●	●	●	●	●	●	●
		14	●	●	○	●	●	●	●	●	○	○	●	●	●	●	●	●	●	●	●
		15	●	○	○	○	○	●	○	○	○	○	●	○	●	●	●	●	●	●	●
		16	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
		17	●	●	●	○	○	●	○	○	○	○	○	○	●	●	●	●	●	●	●

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'd'	STRING	-	-	1	1	ID Configure : ACTION_PLANNING		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|41|W|d|10|1|8|6|6|1|0|0|0000000000000000000|*5E59
ACK Respond	$ESP_OK|49|W|d|1|(ID : ACTION_PLANNING) : Saved Success .|*6A14
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - INTERNET

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'e'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : INTERNET
Number of field values : 8

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	24	0	-	-	Type Internet	
PL3	INTEGER	1	3	1	-	-	Number WiFi in List	
PL4	PRINT	-	-	TEWM_B31B6B	1	30	SSID1	
PL5	PASS_A	-	-	884744F797	0	30	Pass1	
PL6	PRINT	-	-	Tran Trong Toan	1	30	SSID2	
PL7	PASS_A	-	-	12121975	0	30	Pass2	
PL8	PRINT	-	-	My Router	1	30	SSID3	
PL9	PASS_A	-	-		0	30	Pass3	

PL0	PL1	PL2	Type Internet	Unit
'W'	'e'	0	WIFI	
		1	Ethernet ENC28J60	
		2	Ethernet W5500	
		3	Ethernet W5110	
		4	SIM800	
		5	SIM808-SIM868	
		6	SIM900-SIM968-SIM968	
		7	SIM7000	
		8	SIM5320-SIM5360	
		9	SIM7100	
		10	SIM7600	

		11	Ublox LEON-G100	
		12	Ublox LISA-U2xx	
		13	Ublox SARA-G3xx	
		14	Ublox TOBY-L2xx	
		15	Ublox LARA-R2xx	
		16	Ublox MPCI-L2xx	
		17	Ublox SARA-U2xx	
		18	Ublox SARA-R4xx	
		19	Ublox SARA-N4xx	
		20	Quectel BG96	
		21	Quectel M95	
		22	Quectel MC60	
		23	Digi XBee	
		24	SequansMonarch	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'e'	STRING	-	-	1	1	ID Configure : INTERNET		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|67|W|e|0|1|TEWM_B31B6B|884744F797|Tran Trong Toan|12121975|My Router||*190B
ACK Respond	$ESP_OK|42|W|e|1|(ID : INTERNET) : Saved Success .|*271A
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - RADIO_RTK

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'f'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : RADIO_RTK
Number of field values : 10

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	PASS_D	-	-	abc_LoRa_WiFi_012345	1	32	Radio KEY Encrypt	
PL3	INTEGER	0	1	0	-	-	RF Type	
PL4	INTEGER	0	13	0	-	-	WiFi Channel	
PL5	INTEGER	0	9	9	-	-	WiFi Type Power	
PL6	INTEGER	0	5	0	-	-	WiFi Type Speed	
PL7	INTEGER	400000000	1000000000	433000000	-	-	LoRa Frequency(Hz)	Hz
PL8	INTEGER	0	13	0	-	-	LoRa Channel	
PL9	INTEGER	0	10	9	-	-	LoRa Type Power	
PL10	INTEGER	0	8	8	-	-	LoRa Type Speed	
PL11	INTEGER	0	1000000	10	-	-	LoRa delay each 250 byte	Millis second
PL12								
PL13								
PL14								
PL15								
PL16								

PL0	PL1	PL3	RF Type	Unit
'W'	'f'	0	WiFi-to-WiFi	
		1	LoRa-to-LoRa	

PL0	PL1	PL4	WiFi Channel	Unit
PL0	PL1	PL8	LoRa Channel	
'W'	'f'	0	Channel 1	
		1	Channel 2	
		2	Channel 3	
		3	Channel 4	
		4	Channel 5	
		5	Channel 6	
		6	Channel 7	
		7	Channel 8	
		8	Channel 9	
		9	Channel 10	
		10	Channel 11	
		11	Channel 12	
		12	Channel 13	
		13	Channel 14	

PL0	PL1	PL5	WiFi Type Power	Unit
PL0	PL1	PL9	LoRa Type Power	

'W'	'f'	0	2 dBm - 1.59 mW	
		1	4 dBm - 2.51 mW	
		2	6 dBm - 3.98 mW	
		3	8 dBm - 6.31 mW	
		4	10 dBm - 10 mW	
		5	12 dBm - 15.9 mW	
		6	14 dBm - 25.1 mW	
		7	16 dBm - 39.8 mW	
		8	18 dBm - 63.1 mW	
		9	20 dBm - 100 mW	
		10	27 dBm - 500 mW	

PL0	PL1	PL6	WiFi Type Speed	Unit
'W'	'f'		55 Kbps	
			90 Kbps	
			160 Kbps	
			240 Kbps	
			320 Kbps	
			400 Kbps	

PL0	PL1	PL10	LoRa Type Speed	Unit
'W'	'f'		0.15 Kbps	
			0.3 Kbps	

			0.6 Kbps	
			1.2 Kbps	
			2.2 Kbps	
			3.9 Kbps	
			7 Kbps	
			12.5 Kbps	
			22 Kbps	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'f'	STRING	-	-	1	1	ID Configure : RADIO_RTK		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|52|W|f|abc_LoRa_WiFi_012345|0|0|9|0|433000000|0|9|8|10|*1906
ACK Respond	$ESP_OK|43|W|f|1|(ID : RADIO_RTK) : Saved Success .|*7D4A
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - MQTT_RTK

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'g'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : MQTT_RTK
Number of field values : 10

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	STRING	-	-	broker.hivemq.com	1	50	IP	
PL3	INTEGER	0	65535	1883	-	-	Port	
PL4	PASS_A	-	-	rkdpcppo	1	30	User Name	
PL5	PASS_B	-	-	nvZv101KE_SX	1	30	User Password	
PL6	STRING	-	-	ESPRTK_[ID]	1	30	MQTT Client ID	
PL7	STRING	-	-	RTCM	1	50	RTCM Topic	
PL8	STRING	-	-	NMEA	1	50	NMEA topic	
PL9	STRING	-	-	FAST_NMEA	1	50	Fast NMEA topic	
PL10	INTEGER	0	1	1	-	-	Enable encrypt data send	
PL11	PASS_D	-	-	abc_MQTT_RTK_012345	1	32	MQTT KEY Encrypt	
PL12								
PL13								
PL14								
PL15								
PL16								

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'g'	STRING	-	-	1	1	ID Configure : MQTT_RTK		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	

							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|110|W|g|broker.hivemq.com|1883|rkdpcppo|nvZv101KE_SX|ESPRTK_TS9PC_S9SUB|RTCM|NMEA|FAST_NMEA|1|abc_MQTT_RTK_012345|*2D09
ACK Respond	$ESP_OK|42|W|g|1|(ID : MQTT_RTK) : Saved Success .|*3002
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - NTRIP_CLIENT

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'h'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : NTRIP_CLIENT
Number of field values : 6

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	STRING	-	-	rtk2go.com	1	50	IP	
PL3	INTEGER	0	65535	2101	-	-	Port	
PL4	STRING	-	-	no	1	100	Mountpoint	
PL5	PASS_B	-	-	no	1	50	Username-ID	
PL6	PASS_B	-	-	no	1	50	Password	
PL7	INTEGER	0	1	1	-	-	Enable check Mountpoint exist	
PL8								
PL9								
PL10								

PL0	PL1	PL7	Enable check Mountpoint exist	Unit
'W'	'h'	0	Disable check Mountpoint exist	
		1	Enable check Mountpoint exist	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'h'	STRING	-	-	1	1	ID Configure : NTRIP_CLIENT		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|31|W|h|rtk2go.com|2101|no|no|no|1|*0A78
ACK Respond	$ESP_OK|46|W|h|1|(ID : NTRIP_CLIENT) : Saved Success .|*2218
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - NTRIP_MASTER

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'i'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : NTRIP_MASTER
Number of field values : 31

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit

PL2	STRING	-	-	rtk2go.com	1	50	IP	
PL3	INTEGER	0	65535	2101	-	-	Port	
PL4	STRING	-	-	A00__[ID]	1	100	Mountpoint	
PL5	PASS_B	-	-	BETATEST2	1	50	Password	
PL6	INTEGER	0	1	1	-	-	Enable check Mountpoint exist	
PL7	INTEGER	0	1	0	-	-	Enable Use Custom ENTRY	
PL8	PRINT	-	-	San_Jose-Calif	1	30	City	
PL9	INTEGER	0	46	13	-	-	Data Format ID	
PL10	STRING	-	-	1005(1),1077(1),1107(1),1117(1)	1	200	Message type detail	
PL11	INTEGER	0	5	1	-	-	Carrier Phase ID	
PL12	INTEGER	0	1	1	-	-	Flag Beidou	
PL13	INTEGER	0	1	1	-	-	Flag GPS	
PL14	INTEGER	0	1	1	-	-	Flag Glonass	
PL15	INTEGER	0	1	0	-	-	Flag Galileo	
PL16	INTEGER	0	1	1	-	-	Flag QZSS	
PL17	INTEGER	0	1	0	-	-	Flag IRNSS	
PL18	INTEGER	0	1	1	-	-	Flag SBAS	
PL19	INTEGER	0	1	0	-	-	Flag Navic	
PL20	INTEGER	0	1	0	-	-	Flag GNSS	
PL21	INTEGER	0	1	0	-	-	Flag GFZ	
PL22	PRINT	-	-	SNIP	1	20	Network	
PL23	STRING	-	-	USA	1	3	Country (max 3 digit)	
PL24	FLOAT	-90.0	90.0	37.388	-	-	Latitude	deg
PL25	FLOAT	-180.0	180.0	-121.9	-	-	Longitude	deg
PL26	INTEGER	0	1	1	-	-	Request GGA	
PL27	INTEGER	0	1	0	-	-	Solution	
PL28	INTEGER	0	1	0	-	-	Compression/Encryption	
PL29	INTEGER	0	2	1	-	-	Authentication	
PL30	INTEGER	0	1	0	-	-	User Fee	
PL31	INTEGER	0	1000000000	8600	-	-	Speed (Bitrate)	bps
PL32	STRING	-	-	ESPrtk-Base	1	50	Misc	

PL0	PL1	PL6	Enable check Mountpoint exist	Unit
'W'	'i'	0	Dissable check Mountpoint exist	
		1	Enable check Mountpoint exist	

PL0	PL1	PL7	Enable Use Custom ENTRY	Unit
'W'	'i'	0	Disable Use Custom ENTRY	
		1	Enable Use Custom ENTRY	

PL0	PL1	PL9	Data Format ID	PL0	PL1	PL9	Data Format ID
'W'	'i'	0	Auto Detec Format	24			SAE J2735 DSRC
		1	ADV	25			Trimble RMC
		2	AOA	26			Trimble RMC+
		3	BINEX	27			Trimble RMCx
		4	DGPS	28			Trimble sRMCx
		5	RAW	29			Trimble R17
		6	RTCM 2.0	30			Trimble R27
		7	RTCM 2.1	31			Septentrio SBF
		8	RTCM 2.2	32			SAPOS
		9	RTCM 2.3	33			RINEX
		10	RTCM 2.x	34			SP3
		11	RTCM 3.0	35			RTCA
		12	RTCM 3.1	36			FKP
		13	RTCM 3.2	37			COP
		14	RTCM 3.3	38			LB2
		15	RTCM 3.4	39			ZERO
		16	RTCM 3.5	40			POS_ASCII
		17	RTCM 3.x	41			JDNCT
		18	UBlox RAW	42			SP3-ASC
		19	Hermisphere Eclipse	43			UPD-ASC
		20	Skytraq Navspark RAW	44			UPD-RTCM
		21	NMEA	45			ION-ASC

		22	NCT	46			Other Format
		23	Novatel OEM				

PL0	PL1	PL26	Request GGA	Unit
'W'	'i'	0	None	
		1	Yes	

PL0	PL1	PL27	Solution	Unit
'W'	'i'	0	Single reference station	
		1	Networked reference stations	

PL0	PL1	PL28	Compression/Encryption	Unit
'W'	'i'	0	None	
		1	Yes	

PL0	PL1	PL29	Authentication	Unit
'W'	'i'	0	None	
		1	Basic	
		2	Digest	

PL0	PL1	PL30	User Fee	Unit
'W'	'i '	0	No user fee	
		1	Usage is charged	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'i'	STRING	-	-	1	1	ID Configure : NTRIP_MASTER		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|173|W|i|rtk2go.com|2101|A00__81909_GHB9A|BETATEST2|1|0|San_Jose-Calif|13|1005(1),1077(1),1107(1),1117(1)|1|1|1|1|0|1|0|1|0|0|0|SNIP|USA|37.388|-121.9|1|0|0|1|0|8600|ESPrtk-Base|*4C1B
ACK Respond	$ESP_OK|46|W|i|1|(ID : NTRIP_MASTER) : Saved Success .|*2607
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - UART_CONFIGURE

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'j'	Infor 1	Infor 2	Infor 3	Infor 4	...

Note: This configure only work on WEB Configure interface !

Name CFG : UART_CONFIGURE
Number of field values : 19

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	1	1	-	-	Enable Hardware Control	
PL3	INTEGER	0	1	1	-	-	Enable UART Configure	

PL4	INTEGER	0	2	2	-	-	Enable Encode UART Configure	
PL5	INTEGER	0	255	0	-	-	Key Encrypt K_0	
PL6	INTEGER	0	255	1	-	-	Key Encrypt K_1	
PL7	INTEGER	0	255	2	-	-	Key Encrypt K_2	
PL8	INTEGER	0	255	3	-	-	Key Encrypt K_3	
PL9	INTEGER	0	255	4	-	-	Key Encrypt K_4	
PL10	INTEGER	0	255	5	-	-	Key Encrypt K_5	
PL11	INTEGER	0	255	6	-	-	Key Encrypt K_6	
PL12	INTEGER	0	255	7	-	-	Key Encrypt K_7	
PL13	INTEGER	0	255	8	-	-	Key Encrypt K_8	
PL14	INTEGER	0	255	9	-	-	Key Encrypt K_9	
PL15	INTEGER	0	255	10	-	-	Key Encrypt K_10	
PL16	INTEGER	0	255	11	-	-	Key Encrypt K_11	
PL17	INTEGER	0	255	12	-	-	Key Encrypt K_12	
PL18	INTEGER	0	255	13	-	-	Key Encrypt K_13	
PL19	INTEGER	0	255	14	-	-	Key Encrypt K_14	
PL20	INTEGER	0	255	15	-	-	Key Encrypt K_15	

PL0	PL1	PL2	Enable Hardware Control	Unit
'W'	'j'	0	Disable use Hardware Control in UART Configure mode	
		1	Enable use Hardware Control in UART Configure mode	

PL0	PL1	PL3	Enable UART Configure	Unit
'W'	'j'	0	Disable start UART Configure mode	
		1	Enable start UART Configure mode	

PL0	PL1	PL4	Enable Encode UART Configure	Unit
'W'	'j'	0	Disable encrypt data repond Read/Write configure UART Configure	
		1	Enable encrypt data repond Read/Write configure UART Configure. Use Bitwise Encrypt algorithm.	
		2	Enable encrypt data repond Read/Write configure UART Configure. Use AES256 Encrypt algorithm.	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'j'	STRING	-	-	1	1	ID Configure : UART_CONFIGURE		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Note: This configure only work on WEB Configure interface !

Message :	
HOST Send	$ESP_OK|48|W|j|1|1|0|0|1|2|3|4|5|6|7|8|9|10|11|12|13|14|15|*6008

ACK Respond	$ESP_OK|48|W|j|1|(ID : UART_CONFIGURE) : Saved Success .|*3C54
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - BLUETOOTH

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'k'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : BLUETOOTH
Number of field values : 3

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	1	0	-	-	Type Bluetooth	
PL3	PRINT	-	-	BTU-ESPrtk : [ID]	1	100	Bluetooth Classic Name	
PL4	PRINT	-	-	BLE-ESPrtk : [ID]	1	100	Bluetooth BLE Name	

PL0	PL1	PL2	Type Bluetooth	Unit
'W'	'k'	0	Bluetooth Classic - BTU	
		1	Bluetooth Low Energy - BLE	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'k'	STRING	-	-	1	1	ID Configure : BLUETOOTH		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|56|W|k|0|BTU-ESPrtk : 81909_GHB9A|BLE-ESPrtk : 81909_GHB9A|*6632

ACK Respond	$ESP_OK|43|W|k|1|(ID : BLUETOOTH) : Saved Success .|*6556
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - DISPLAY_VIEWER

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'I'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : DISPLAY_VIEWER
Number of field values : 8

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	2	1	-	-	Type Display	
PL3	INTEGER	0	6	6	-	-	Brightness Display	
PL4	INTEGER	0	6	6	-	-	Contrast Display	
PL5	INTEGER	0	1	0	-	-	Flip Display	
PL6	INTEGER	0	1	1	-	-	Use ERROR-LED	
PL7	INTEGER	0	1	0	-	-	Invert ERROR-LED state	
PL8	INTEGER	0	1	1	-	-	Use CONNECTION-LED	
PL9	INTEGER	0	1	0	-	-	Invert CONNECTION-LED state	

PL0	PL1	PL	Type Display	Unit
'W'	'I'	0	NEOPIXEL LED (4 LED)	
		1	Oled SSD1306 0.96" (I2C - 128x64)	
		2	Oled SH1106 1.3" (I2C - 128x64)	

PL0	PL1	PL3	Brightness Display	Unit
PL0	PL1	PL4	Contrast Display	
'W'	'I'	0	10	
		1	20	
		2	50	
		3	100	
		4	150	

		5	200	
		6	255	

PL0	PL1	PL5	Flip Display	Unit
'W'	'l'	0	No Flip Display .	
		1	Yes Flip Display. If type display NEOPIXEL LED: Rorate index neopixel 1234 to 4321 If type display OLED Display: Vertical flip screen .	

PL0	PL1	PL6	Use ERROR-LED	Unit
PL0	PL1	PL8	Use CONNECTION-LED	
'W'	'l'	0	Not use	
		1	Use	

PL0	PL1	PL7	Invert ERROR-LED state	Unit
PL0	PL1	PL9	Invert CONNECTION-LED state	
'W'	'l'	0	No Invert	
		1	Yes Invert	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'l'	STRING	-	-	1	1	ID Configure : DISPLAY_VIEWER		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|20|W|l|1|6|6|0|1|0|1|0|*2509
ACK Respond	$ESP_OK|48|W|l|1|(ID : DISPLAY_VIEWER) : Saved Success .|*2044
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - EVENT_LOG

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'm'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : EVENT_LOG
Number of field values : 5

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	9	3	-	-	Type Size Event_Log File	
PL3	INTEGER	0	1	0	-	-	Enable Record Action	
PL4	INTEGER	0	1	0	-	-	Enable Record Error	
PL5	INTEGER	0	1	0	-	-	Enable Record Action Planing	
PL6	INTEGER	0	1	0	-	-	Enable Record WiFi Scan	

PL0	PL1	PL2	Type Size Event_Log File	Unit
'W'	'm'	0	5 Kbytes	
		1	10 Kbytes	
		2	15 Kbytes	
		3	20 Kbytes	
		4	25 Kbytes	
		5	30 Kbytes	
		6	35 Kbytes	
		7	40 Kbytes	

PL0	PL1	PL3	Enable Record Action	Unit
PL0	PL1	PL4	Enable Record Error	
PL0	PL1	PL5	Enable Record Action Planing	
PL0	PL1	PL6	Enable Record WiFi Scan	
'W'	'm'	0	Enalbe	
		1	Disable	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'm'	STRING	-	-	1	1	ID Configure : EVENT_LOG		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|14|W|m|3|0|0|0|0|*6D76
ACK Respond	$ESP_OK|43|W|m|1|(ID : EVENT_LOG) : Saved Success .|*624A
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - SD_CARD

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'o'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : SD_CARD
Number of field values : 7

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	STRING	-	-	my_record	3	20	File's Name	
PL3	INTEGER	0	9	0	-	-	Format File	
PL4	INTEGER	0	1	0	-	-	Type write	
PL5	INTEGER	10	4000000	500	-	-	MAX_SIZE_FILE (K Byte)	K Byte
PL6	INTEGER	0	100000000	1	-	-	RTK-Rover NMEA Record Cycle	second
PL7	INTEGER	1	65000	2	-	-	RTK-Rover Max NMEA save each Update Cycle	
PL8	BIT	-	-	11000000000000000000000000000000	32	32	RTK-Rover Enable type NMEA record	

PL0	PL1	PL3	Format File	Unit
'W'	'o'	0	.log	
		1	.txt	
		2	.nmea	
		3	.nme	
		4	.gps	
		5	.ubx	
		6	data	
		7	.dat	
		8	.bin	
		9	(No type): User type format at the end file's name like (my_record.txt)	

PL0	PL1	PL4	Type write	Unit
'W'	'o'	0	Append to old file	
		1	Create and write on new file	

PL0	PL1	PL8	RTK-Rover Enable type NMEA record	PL0	PL1	PL	RTK-Rover Enable type NMEA record
'W'	'o'	BIT1	NMEA - GGA	'W'	''	BIT16	NMEA - BWC
		BIT2	NMEA - RMC			BIT17	NMEA - MSK
		BIT3	NMEA - GSA			BIT18	NMEA - MSS
		BIT4	NMEA - GSV			BIT19	NMEA - RMA
		BIT4	NMEA - VTG			BIT20	NMEA - RMB
		BIT5	NMEA - GLL			BIT21	NMEA - RTE
		BIT6	NMEA - ZDA			BIT22	NMEA - TRF
		BIT7	NMEA - GNS			BIT23	NMEA - STN
		BIT8	NMEA - GST			BIT24	NMEA - VBW
		BIT9	NMEA - DTM			BIT25	NMEA - WCV
		BIT10	NMEA - GBS			BIT26	NMEA - WPL
		BIT11	NMEA - GRS			BIT27	NMEA - XTC
		BIT12	NMEA - AAM			BIT28	NMEA - XTE

		BIT13	NMEA - ALM			BIT29	NMEA - ZTG
		BIT14	NMEA - APA			BIT30	Other Type such as RTCM ,RAW ,UBX , STI ..
		BIT15	NMEA - APB			BIT31	Reserve – default set to 0
		BIT16	NMEA - BOD			BIT32	Reserve – default set to 0

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'o'	STRING	-	-	1	1	ID Configure : SD_CARD		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|59|W|o|my_record|0|0|500|1|2|11000000000000000000000000000000|*2500
ACK Respond	$ESP_OK|41|W|o|1|(ID : SD_CARD) : Saved Success .|*6917
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - IMU_MPU9250

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'p'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : IMU_MPU9250
Number of field values : 19

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	9	1	-	-	Type Update frequency	
PL3	INTEGER	0	3	3	-	-	Gyroscope Scale Range	
PL4	INTEGER	0	3	1	-	-	Accelerometer Scale Range	
PL5	INTEGER	0	6	0	-	-	Low pass Filter	
PL6	FLOAT	1.00	20.0	4.512	-	-	Shock Kalman Filter	
PL7	BIT	-	-	000000111	9	9	Enable/Disable data output	
PL8	BIT	-	-	00001111	8	8	Enable Apply Extended Kalman filter	
PL9	FLOAT	-1000000000.0	1000000000.0	0	-	-	Matrix Convert - MBx	
PL10	FLOAT	-1000000000.0	1000000000.0	0	-	-	Matrix Convert - MBy	
PL11	FLOAT	-1000000000.0	1000000000.0	0	-	-	Matrix Convert - MBz	
PL12	FLOAT	-1000000000.0	1000000000.0	1	-	-	Matrix Convert - M11	
PL13	FLOAT	-1000000000.0	1000000000.0	0	-	-	Matrix Convert - M12	
PL14	FLOAT	-1000000000.0	1000000000.0	0	-	-	Matrix Convert - M13	
PL15	FLOAT	-1000000000.0	1000000000.0	0	-	-	Matrix Convert - M21	
PL16	FLOAT	-1000000000.0	1000000000.0	1	-	-	Matrix Convert - M22	
PL17	FLOAT	-1000000000.0	1000000000.0	0	-	-	Matrix Convert - M23	
PL18	FLOAT	-1000000000.0	1000000000.0	0	-	-	Matrix Convert - M31	
PL19	FLOAT	-1000000000.0	1000000000.0	0	-	-	Matrix Convert - M32	
PL20	FLOAT	-1000000000.0	1000000000.0	1	-	-	Matrix Convert - M33	

PL0	PL1	PL2	Type Update frequency	Unit
'W'	'p'	0	2 Hz	
		1	10 Hz	
		2	15 Hz	
		3	20 Hz	
		4	25 Hz	
		5	35 Hz	
		6	50 Hz	
		7	100 Hz	
		8	150 Hz	
		9	200 Hz	

PL0	PL1	PL3	Gyroscope Scale Range	Unit
'W'	'p'	0	250	
		1	500	
		2	1000	
		3	2000	

PL0	PL1	PL4	Accelerometer Scale Range	Unit
'W'	'p'	0	2	
		1	4	
		2	8	
		3	16	

PL0	PL1	PL5	Low pass Filter	Unit
'W'	'p'	0	Not use	
		1	5 Hz	
		2	10 Hz	
		3	20 Hz	
		4	41 Hz	
		5	92 Hz	

		6	184 Hz	

PL0	PL1	PL7	Enable/Disable data output IM1 $ESPIM1,GX,GY,GZ,MX,MY,MZ,AX,AY,AZ,T,MXc,MYc,MZc,*xx	Unit
'W'	'p'	BIT1	Raw Gyroscope 3 Axis(GX,GY,GZ)	
		BIT2	Raw Magnetometer 3 Axis(MX,MY,MZ)	
		BIT3	Raw Accelerometer 3 Axis(AX,AY,AZ)	
		BIT4	Temperature(T)	
		BIT5	Magnetometer 3 Axis Calibrated(MXc,MYc,MZc)	
			Enable/Disable data output IM2 $ESPIM2,QW,QX,QY,QZ,Yaw,Pitch,Roll,H,FQW,FQX,FQY,FQZ,Fyaw,Fpitch,Froll,FH,*xx	Unit
		BIT6	DMP-Quaternion (QW,QX,QY,QZ)	
		BIT7	DMP-Euler 3 Axis(Yaw,Pitch,Roll)	
		BIT8	Compass Heading : Calibrated + Tilt compensated(H)	
		BIT9	Data Filter (FQW,FQX,FQY,FQZ,Fyaw,Fpitch,Froll,FH)	

PL0	PL1	PL8	**Enable Apply Extended Kalman filter**	**Unit**
'W'	'p'	BIT1	Filter QW	
		BIT2	Filter QX	
		BIT3	Filter QY	
		BIT4	Filter QZ	
		BIT5	Filter Yaw	
		BIT6	Filter Pitch	
		BIT7	Filter Roll	
		BIT8	Filter Heading	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'p'	STRING	-	-	1	1	ID Configure : IMU_MPU9250	
	PL3	INTEGER	0	1	-	-	Status Configure	

							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|61|W|p|1|3|1|0|4.512|000000111|00001111|0|0|0|1|0|0|0|1|0|0|0|1|*5E4C
ACK Respond	$ESP_OK|45|W|p|1|(ID : IMU_MPU9250) : Saved Success .|*6617
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - GPIO_PIN_MAP

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'q'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : GPIO_PIN_MAP
Number of field values : 26

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	ID_ESPrtk_Pin map to GPIOx	Unit
PL2	INTEGER	0	25	18	-	-	GPIO 0 ← ID_ESPrtk_Pin	
PL3	INTEGER	0	25	4	-	-	GPIO 1 ← ID_ESPrtk_Pin	
PL4	INTEGER	0	25	21	-	-	GPIO 2 ← ID_ESPrtk_Pin	
PL5	INTEGER	0	25	1	-	-	GPIO 3 ← ID_ESPrtk_Pin	
PL6	INTEGER	0	25	16	-	-	GPIO 4 ← ID_ESPrtk_Pin	
PL7	INTEGER	0	25	10	-	-	GPIO 5 ← ID_ESPrtk_Pin	
PL8	INTEGER	0	25	14	-	-	GPIO 12 ← ID_ESPrtk_Pin	
PL9	INTEGER	0	25	0	-	-	GPIO 13 ← ID_ESPrtk_Pin	
PL10	INTEGER	0	25	17	-	-	GPIO 14 ← ID_ESPrtk_Pin	
PL11	INTEGER	0	25	11	-	-	GPIO 15 ← ID_ESPrtk_Pin	
PL12	INTEGER	0	25	13	-	-	GPIO 16 ← ID_ESPrtk_Pin	
PL13	INTEGER	0	25	7	-	-	GPIO 17 ← ID_ESPrtk_Pin	
PL14	INTEGER	0	25	8	-	-	GPIO 18 ← ID_ESPrtk_Pin	
PL15	INTEGER	0	25	2	-	-	GPIO 19 ← ID_ESPrtk_Pin	
PL16	INTEGER	0	25	22	-	-	GPIO 21 ← ID_ESPrtk_Pin	
PL17	INTEGER	0	25	19	-	-	GPIO 22 ← ID_ESPrtk_Pin	
PL18	INTEGER	0	25	12	-	-	GPIO 23 ← ID_ESPrtk_Pin	
PL19	INTEGER	0	25	9	-	-	GPIO 25 ← ID_ESPrtk_Pin	
PL20	INTEGER	0	25	23	-	-	GPIO 26 ← ID_ESPrtk_Pin	
PL21	INTEGER	0	25	20	-	-	GPIO 27 ← ID_ESPrtk_Pin	
PL22	INTEGER	0	25	15	-	-	GPIO 32 ← ID_ESPrtk_Pin	

PL23	INTEGER	0	25	3	-	-	GPIO 33 ← ID_ESPrtk_Pin	
PL24	INTEGER	0	25	24	-	-	GPIO 34 ← ID_ESPrtk_Pin	
PL25	INTEGER	0	25	6	-	-	GPIO 35 ← ID_ESPrtk_Pin	
PL26	INTEGER	0	25	25	-	-	GPIO 36 ← ID_ESPrtk_Pin	
PL27	INTEGER	0	25	5	-	-	GPIO 39 ← ID_ESPrtk_Pin	

ID_ESPrtk_Pin	Name pin	ID_ESPrtk_Pin	Name pin
0	CWB_Button	13	RX_UART2
1	RX_UART0	14	ERROR_LED
2	MOSI_SD_Card	15	CS_LoRa
3	MOSI_LoRa_Ethe	16	TX_UART2
4	TX_UART0	17	TX_UART3
5	MISO_SD_Card	18	CS_Ethernet__PWR_GSM
6	MISO_LoRa_Ethe	19	SCL_I2C
7	RX_UART1	20	SCL_OLED
8	SCK_SD_Card	21	RST_LoRa_Ethe
9	SCK_LoRa_Ethe	22	SDA_I2C
10	TX_UART1	23	SDA_OLED
11	CONNECT_LED	24	DI0_LoRa
12	CS_SD_Card	25	RX_UART3

Example

Example 1	JSON - Message Configure
HOST Send	$ESP_OK|72|W|q|18|4|21|1|16|10|14|0|17|11|13|7|8|2|22|19|12|9|23|20|15|3|24|6|25|5|*3F0B
Default from factory	{ "CWB_Button" : "13" , "RX_UART0" : "3" , "MOSI_SD_Card" : "23" , "MOSI_LoRa_Ethe" : "33" , "TX_UART0" : "1" , "MISO_SD_Card" : "39" , "MISO_LoRa_Ethe" : "35" , "RX_UART1" : "17" , "SCK_SD_Card" : "18" , "SCK_LoRa_Ethe" : "25" , "TX_UART1" : "5" , "CONNECT_LED" : "15" , "CS_SD_Card" : "19" , "RX_UART2" : "16" , "ERROR_LED" : "12" , "CS_LoRa" : "32" , "TX_UART2" : "4" , "TX_UART3" : "14" , "CS_Ethernet__PWR_GSM" : "0" , "SCL_I2C" : "22" "SCL_OLED" : "27" , "RST_LoRa_Ethe" : "2" , "SDA_I2C" : "21" , "SDA_OLED" : "26" , "DI0_LoRa" : "34" , "RX_UART3" : "36" }
Example 2	**JSON - Message Configure**
HOST Send	$ESP_OK|72|W|q|18|4|5|1|10|9|16|12|8|2|17|11|15|6|23|20|21|14|24|3|22|19|7|13|0|25|*3F4F
Example configure of User	{ "CWB_Button" : "36" , "RX_UART0" : "3" , "MOSI_SD_Card" : "15" , "MOSI_LoRa_Ethe" : "27" , "TX_UART0" : "1" , "MISO_SD_Card" : "2" , "MISO_LoRa_Ethe" : "19" , "RX_UART1" : "34" , "SCK_SD_Card" : "14" , "SCK_LoRa_Ethe" : "5" , "TX_UART1" : "4" , "CONNECT_LED" : "17" , "CS_SD_Card" : "13" , "RX_UART2" : "35" , "ERROR_LED" : "25" , "CS_LoRa" : "18" , "TX_UART2" : "12" , "TX_UART3" : "16" , "CS_Ethernet__PWR_GSM" : "0" , "SCL_I2C" : "33" , "SCL_OLED" : "22" , "RST_LoRa_Ethe" : "23" , "SDA_I2C" : "32" , "SDA_OLED" : "21" , "DI0_LoRa" : "26" , "RX_UART3" : "39" }
ACK Respond	$ESP_OK|46|W|q|1|(ID : GPIO_PIN_MAP) : Saved Success .|*3616
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	‘q’	STRING	-	-	1	1	ID Configure : GPIO_PIN_MAP	

	PL3	INTEGER	0	1	-	-	Status Configure	
							0	Error: Failed to save , please try again
							1	OK : Saved configure success
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format	

Command Write Configure - SYSTEM_Other_Mode

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'r'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : SYSTEM_Other_Mode
Number of field values : 9

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	5	0	-	-	Type CHIP ESP32	
PL3	INTEGER	0	9	9	-	-	Power System Saving	
PL4	INTEGER	0	7	3	-	-	Type speed SPI-SD CARD	
PL5	INTEGER	0	7	3	-	-	Type speed SPI-ETHERNET	
PL6	INTEGER	0	7	3	-	-	Type speed SPI-LORA	
PL7	INTEGER	0	7	3	-	-	Type speed I2C-IMU	
PL8	INTEGER	0	7	3	-	-	Type speed I2C-OLED	
PL9	BIT	-	-	111111111111	12	12	Enable print message types on TX_UART0 port	
PL10	BIT	-	-	11111111111111111111	20	20	Enable warning to users when an error occurs	

PL0	PL1	PL2	Type CHIP ESP32	Unit
'W'	'r'	0	ESP32_D0WD	
		1	ESP32_D0WDQ6	
		2	ESP32_D0WD_V3	
		3	ESP32_D0WDQ6_V3	
		4	ESP32_PICO_D4	
		5	ESP32_PICO_V3	

PL0	PL1	PL3	Power System Saving	Unit
'W'	'r'	0	10% of Max Power	
		1	20% of Max Power	
		2	30% of Max Power	
		3	40% of Max Power	
		4	50% of Max Power	
		5	60% of Max Power	
		6	70% of Max Power	
		7	80% of Max Power	
		8	90% of Max Power	
		9	100% of Max Power	

PL0	PL1	PL4	Type speed SPI-SD CARD	Unit
		PL5	Type speed SPI-ETHERNET	
		PL6	Type speed SPI-LORA	
		PL7	Type speed I2C-IMU	
		PL8	Type speed I2C-OLED	
'W'	'r'	0	(Lowest speed) 25% of Standard Speed	
		1	50% of Standard Speed	
		2	75% of Standard Speed	
		3	100% of Standard Speed	
		4	125% of Standard Speed	
		5	150% of Standard Speed	
		6	175% of Standard Speed	

		7	(Highest speed) 200% of Standard Speed	

PL0	PL1	PL9	Enable print message types on TX_UART0 port	Unit
'W'	'r'	BIT1	_SYSTEM_ (ESPrtk System)	
		BIT2	_ERROR_ (Error status)	
		BIT3	_INFO_ (Text , Log, Information...)	
		BIT4	_GOIN_ (Go to function status)	
		BIT5	_WFSCAN_ (Wifi Scan Result)	
		BIT6	_BROKER_ (Broker MQTT Status)|	
		BIT7	_RVNMEA_ (print NMEA from Rover)|	
		BIT8	_CONFIG_ (Configure information detail)	
		BIT9	$__GGA (Rover print GGA message)	
		BIT10	$_NMEA_ (Rover print NMEA message)	
		BIT11	$ESPIM1 (IMU MPU9250 Data type 1)	
		BIT12	$ESPIM2 (IMU MPU9250 Data type 2)	

PL0	PL1	PL10	Enable warning to users when an error occurs	Unit
'W'	'r'	BIT1	FIRST SETTING	
		BIT2	UART	
		BIT3	INTERNET	
		BIT4	MQTT	
		BIT5	NTRIP	
		BIT6	RADIO_RF_PHY	
		BIT7	RADIO_RF_ROVER	
		BIT8	BLUETOOTH	
		BIT9	SD_CARD	
		BIT10	IMU_MPU	
		BIT11	DISPLAY_VIEWER	
		BIT12	UART_CONIGURE	
		BIT13	EVENT_LOG	
		BIT14	Other1	
		BIT15	Other2	

		BIT16	Other3	
		BIT17	Other4	
		BIT18	Other5	
		BIT19	Other6	
		BIT20	Other7	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'r'	STRING	-	-	1	1	ID Configure : SYSTEM_Other_Mode		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|52|W|r|0|9|3|3|3|3|3|111111111111|11111111111111111111|*790E
ACK Respond	$ESP_OK|51|W|r|1|(ID : SYSTEM_Other_Mode) : Saved Success .|*4B63
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - GSM_CELLULAR

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	's'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : GSM_CELLULAR
Number of field values : 13

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	1	0	-	-	Type configure GSM	
PL3	INTEGER	0	1	0	-	-	View AT TX-RX	
PL4	PRINT	-	-	1234	1	40	SIM PIN	
PL5	PRINT	-	-	w3-world	1	40	APN network	
PL6	PRINT	-	-	mms	1	40	GPRS Username	
PL7	PRINT	-	-	mms	1	40	GPRS Password	
PL8	INTEGER	0	1	1	-	-	Auto restart GSM	
PL9	INTEGER	0	14	7	-	-	Type Max Baudrate	
PL10	INTEGER	0	21	0	-	-	Net Mode	
PL11	BIT	-	-	10000000000000000000000	23	23	GSM/2G/3G Band	
PL12	BIT	-	-	10000000000000000000000000000000000000000000000000000000000000000000000000000000	66	66	4G/LTE Band	
PL13	BIT	-	-	10000000	8	8	TDS Band	
PL14	AT_CODE	-	-	[RST][COM^9600][DLY^1][ATE0^5^10^OK][AT+CMEE=0^5^10^OK^ok][CLR][PIN^1234][CFG][NET^w3-world^mms^mms][COM^115200]	0	1500	Custom AT Code	

PL0	PL1	PL2	Type configure GSM	Unit
'W'	's'	0	Auto Configure	
		1	Custom AT Configure	

PL0	PL1	PL3	View AT TX-RX	Unit
'W'	's'	0	Disable View AT TX-RX	
		1	Enable View AT TX-RX	

PL0	PL1	PL8	Auto restart GSM	Unit
'W'	's'	0	Disable Auto restart GSM	
		1	Enable Auto restart GSM	

PL0	PL1	PL9	Type Max Baudrate	Unit
'W'	's'	0	1200	bps
		1	2400	bps
		2	4800	bps
		3	9600	bps
		4	19200	bps
		5	38400	bps
		6	57600	bps
		7	115200	bps
		8	230400	bps
		9	250000	bps
		10	460800	bps

		11	500000	bps
		12	921600	bps
		13	1000000	bps
		14	2000000	bps

PL0	PL1	PL10	Net Mode	Unit
'W'	's'	0	AUTO	
		1	GSM only	
		2	UMTS only	
		3	LTE only	
		4	LTE_M1 only	
		5	LTE_NB1 only	
		6	EGPRS only	
		7	WCDMA only	
		8	TDS_CDMA only	
		9	EVDO only	
		10	GSM + EGPRS	
		11	GSM + LTE	
		12	GSM + UMTS	
		13	GSM + WCDMA	
		14	GSM + TDSCDMA	
		15	CDMA + EVDO	
		16	GSM + UMTS + LTE	
		17	GSM + WCDMA + LTE	
		18	GSM + EGPRS + LTE	
		19	GSM + TDSCDMA + LTE	
		20	GSM + WCDMA + TDSCDMA	

PL0	PL1	PL11	GSM/2G/3G Band	Unit
'W'	's'	BIT1	Auto	

		BIT2	Any_Value	
		BIT3	GSM_450	
		BIT4	GSM_480	
		BIT5	GSM_750	
		BIT6	GSM_850	
		BIT7	GSM_900	
		BIT8	GSM_1800	
		BIT9	GSM_1900	
		BIT10	GSM_EGSM_900	
		BIT11	GSM_PGSM_900	
		BIT12	GSM_RGSM_900	
		BIT13	GSM_DCS_1800	
		BIT14	GSM_PCS_1900	
		BIT15	WCDMA_800	
		BIT16	WCDMA_850	
		BIT17	WCDMA_VIII_900	
		BIT18	WCDMA_III_1700	
		BIT19	WCDMA_IV_1700	
		BIT20	WCDMA_IX_1700	
		BIT21	WCDMA_PCS_1900	
		BIT22	WCDMA_IMT_2000	
		BIT23	WCDMA_VII_2600	

PL0	PL1	PL12	4G/LTE Band	Unit
'W'	's'	BIT1	Auto	
		BIT2	Any_Value	
		BIT3	Band 0	
		BIT4	Band 1	
		BIT5	Band 2	
		BIT6	Band 3	
		...		
		BIT X	Band (X-3)	
		...		
		BIT66	Band 63	

PL0	PL1	PL13	TDS Band	Unit
'W'	's'	BIT1	Auto	
		BIT2	Any_Value	
		BIT3	A (1900-1920 MHz, 2010-2020 MHz)	
		BIT4	B (1850-1910 MHz, 1930-1990 MHz)	
		BIT5	C (1910-1930 MHz)	
		BIT6	D (2570-2620 MHz)	
		BIT7	E (2300-2400 MHz)	
		BIT8	F (1880-1920 MHz)	

Type	Custom AT Code	Example
[PWR^a]	Retart GSM device by pull-down PWR pin (in a second) and wait until GSM wakeup + ready.	[PWR^2]
[RST]	Retart/Reset GSM device by AT command and wait until GSM ready.	[RST]
[COM^a]	Try scan to communicate with GSM device at baudrate a bps. If failed, auto select baudrate value and try again.	[COM^9600]
[CLR]	Clear all RX buffer UART3.	[CLR]
[DLY^a]	Delay in a Second.	[DLY^1]
[PIN^abcd]	If SIM locked , set PIN SIM is abcd to un-lock SIM.	[PIN^1234]
[CFG]	Auto send AT to Configure Mode / Band /... for GSM device.	[CFG]
[NET^a^b^c]	Auto send AT to Open Network for GSM device. APN String Network: a GPRS Username: b GPRS Password: c	[NET^w3-world^mms^mms]
[TCP^a^b^c]	Connect to Server IP: a,Port: b . If connect success , wait respond from Server after send string c . (Auto replace \r \n to 0x0D 0x0A as byte data) (Just for Dev)	[TCP^google.com^80^GET /search?q=esprtk HTTP/1.1\r\nHost: www.google.com\r\ nConnection: close\r\n\r\n]
[a^b^c^d]	Send string a Wait until respond contain d Max time wait : b If not receive, try again, max try : c	[ATE0^5^10^OK]
[a^b^c^d1^d2]	Send string a Wait until respond contain d1 or d2 Max time wait : b If not receive, try again, max try : c	[AT+CMEE=0^5^10^OK^ ok]
...		
[a^b^c^d1^d2 ^d3^d4^d5^d6]	Send string a Wait until respond contain d1 or d2 or d3 (or d4 or d5 or d6 .) Max time wait : b If not receive, try again, max try : c	[AT+CMEE=0^5^10^OK^ ok^Ok^Error]

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘s’	STRING	-	-	1	1	ID Configure : GSM_CELLULAR		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|249|W|s|0|0|1234|w3-world|mms|mms|0|7|0|10000000000000000000| 10000000000000000000000000000000000000000000000000000000000000000000000000000|10000000|[RST] [COM^9600][DLY^1][ATE0^5^10^OK][AT+CMEE=0^5^10^OK^ok][CLR][PIN^1234][CFG] [NET^w3-world^mms^mms][COM^115200]|*0F54
ACK Respond	$ESP_OK|46|W|s|1|(ID : GSM_CELLULAR) : Saved Success .|*2401
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - STREAM_MANAGER

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	't'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : STREAM_MANAGER
Number of field values : 7

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	1	0	-	-	Enable STREAM_MANAGER	
PL3	INTEGER	0	1	0	-	-	Enable Print status Push-In	
PL4	INTEGER	0	1	0	-	-	Enable Print status Push-In (Full)	
PL5	INTEGER	0	1	0	-	-	Enable Print status Push-Out	
PL6	INTEGER	0	1	0	-	-	Enable Print status Push-Out (Full)	
PL7	INTEGER	0	100000000	0	-	-	Limit Speed Push-Out	Bytes/ Second
PL8	AT_CODE	-	-		0	1500	Custom AT Code	

PL0	PL1	PL2	Enable STREAM_MANAGER	Unit
'W'	't'	0	Disable STREAM_MANAGER	
		1	Enable STREAM_MANAGER	

PL0	PL1	PL3	Print status Push-In	Unit
'W'	't'	0	Disable Print status Push-In	
		1	Enable Print status Push-In	

PL0	PL1	PL4	Print status Push-In (Full)	Unit
'W'	't'	0	Disable Print status Push-In (Full)	
		1	Enable Print status Push-In (Full)	

PL0	PL1	PL5	Print status Push-Out	Unit
'W'	't'	0	Disable Print status Push-Out	
		1	Enable Print status Push-Out	

PL0	PL1	PL6	Print status Push-Out (Full)	Unit
'W'	't'	0	Disable Print status Push-Out (Full)	
		1	Enable Print status Push-Out (Full)	

	Custom AT Code	Example
[ADD^a^b]	Auto to insert RAW data . Cycle a second. RAW in HEX : b	**[ADD^1^D300063F00000000009925CA]** - Auto add empty RTCM 1008 to stream, cycle 1 second
[T-ID^RMV]	If receive message . Remove this message	**[R-1007^RMV]** - Remove RTCM 1007
[T-ID^ISA^a]	Inser RAW data After T-ID message. RAW in HEX : a	**[R-1008^ISA^0A0B0C0D]** - Insert 4 byte 0A0B0C0D after RTCM-1008 message.
[T-ID^ISB^a]	Insert RAW data Before T-ID message. RAW in HEX :a	**[N-GPGGA^ISB^0A0B0C0D]** - Insert 4 byte 0A0B0C0D before NMEA-$GPGGA message.
[T-ID^RPL^a]	Replace T-ID message by RAW data . RAW in HEX : a	**[R-1074^RPL^0A0B0C0D]** - Replace RTCM-1074 message by 4 byte 0A0B0C0D
[T-ID^RPT^a]	Repeat T-ID message . Max repeat : a	**[N-GPRMC^RPT^3]** -Repeat NMEA-$GPRMC message 3 times.

[T-ID^RDC^a]	Count T-ID message .
	If (Count < a) : Disable this message , set Count=Count+1.
	If (Count >= a) : Enable this message go through stream , reset Count = 0 .
		[R-1005^RDC^3]
		- Reduce RTCM-1005 message 3 times.

T-ID	Type message	Example
R-i	Type = RTCM , ID Integer = i	R-1007 R-1005
N-i	Type =NMEA, Header String= i	N-GPGGA N-GNRMC
U-i1i2	Type =UBX , Class message HEX= i1, ID message HEX = i2	U-0106 U-0A02
S-i	Type =Skytraq , ID message HEX = i	S-01 S-0E

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘t’	STRING	-	-	1	1	ID Configure : STREAM_MANAGER		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|17|W|t|0|0|0|0|0|0||*444E
HOST Send	$ESP_OK|49|W|t|1|1|0|1|0|0|[ADD^5^0A0B0C0D0E][R-1005^RDC^3]|*564E
ACK Respond	$ESP_OK|48|W|t|1|(ID : STREAM_MANAGER) : Saved Success .|*374A
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - POSITION_ANTENNA

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'u'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : POSITION_ANTENNA
Number of field values : 32

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	BIT	-	-	000000000	9	9	Type messages enable	
PL3	INTEGER	0	10000	60	-	-	NMEA GGA repeat cycle	Second
PL4	INTEGER	0	10000	60	-	-	RTCM 1005 repeat cycle	Second
PL5	INTEGER	0	10000	60	-	-	RTCM 1006 repeat cycle	Second
PL6	INTEGER	0	10000	60	-	-	RTCM 1007 repeat cycle	Second
PL7	INTEGER	0	10000	60	-	-	RTCM 1008 repeat cycle	Second
PL8	INTEGER	0	10000	60	-	-	RTCM 1032 repeat cycle	Second
PL9	INTEGER	0	10000	60	-	-	RTCM 1033 repeat cycle	Second
PL10	INTEGER	0	10000	1	-	-	RTCM 1004 repeat cycle	Second
PL11	INTEGER	0	10000	1	-	-	RTCM 1012 repeat cycle	Second
PL12	FLOAT	-90.00	90.00	47.082164	-	-	Rover Latitude|	Deg
PL13	FLOAT	-180.0	180.0	2.399216	-	-	Rover Longitude	Deg
PL14	FLOAT	-1000000	10000000	10.000	-	-	Rover Altitude	Meter
PL15	STRING	-	-	GP	2	2	GGA Talker ID	
PL16	FLOAT	-90.00	90.00	47.000000	-	-	Base Latitude	Deg
PL17	FLOAT	-180.0	180.0	2.000000	-	-	Base Longitude	Deg
PL18	FLOAT	-1000000	10000000	10.000	-	-	Base Altitude	Meter
PL19	FLOAT	0.00	6.5535	0	-	-	Base Antenna Height	Meter
PL20	INTEGER	0	3	0	-	-	Quarter Cycle Indicator	
PL21	INTEGER	0	63	0	-	-	ITRF Realization Year	
PL22	INTEGER	0	4095	0	-	-	Reference Station ID	
PL23	INTEGER	0	4095	0	-	-	Physical Reference Station ID	
PL24	INTEGER	0	255	123	-	-	Antenna Setup ID	

PL25	PRINT	-	-	ADVNULLANTENNA	0	31	Antenna Descriptor	
PL26	PRINT	-	-	0.0.0.0	0	31	Antenna Serial Number	
PL27	PRINT	-	-	ESPRTK	0	31	Receiver Type Descriptor	
PL28	PRINT	-	-	3.9.0 _ 4.x.x	0	31	Receiver Firmware Version	
PL29	PRINT	-	-	[ESPrtk's ID]	0	31	Receiver Serial Number	
PL30	BIT	-	-	10000	5	5	GNSS System Indicator and Single Receiver Oscillator Indicator	
PL31	INTEGER	1	32	20	-	-	Max satellite GPS	
PL32	INTEGER	1	27	20	-	-	Max satellite GLONASS	
PL33	INTEGER	1	35	1	-	-	Max satellite SBAS	
PL34	INTEGER	1	96	96	-	-	Max OBS in an Epoch	

PL0	PL1	PL2	Type messages enable	Unit
'W'	's'	BIT1	Enable auto use default NMEA GGA messsage if GNSS's data not available.	(*)
		BIT2	Enable auto use default RTCM 1005 messsage if GNSS's data not available.	(*)
		BIT3	Enable auto use default RTCM 1006 messsage if GNSS's data not available.	(*)
		BIT4	Enable auto use default RTCM 1007 messsage if GNSS's data not available.	(*)
		BIT5	Enable auto use default RTCM 1008 messsage if GNSS's data not available.	(*)
		BIT6	Enable auto use default RTCM 1032 messsage if GNSS's data not available.	(*)
		BIT7	Enable auto use default RTCM 1033 messsage if GNSS's data not available.	(*)
		BIT8	Enable auto use default RTCM 1004 messsage if GNSS's data not available.	(*)
		BIT9	Enable auto use default RTCM 1012 messsage if GNSS's data not available.	(*)

(*) : BIT = 1 is Enable , BIT = 0 is Disable .

PL0	PL1	PL30	GNSS System Indicator and Single Receiver Oscillator Indicator	Unit
'W'	's'	BIT1	GPS Indicator in use.	(*)
		BIT2	GLONASS Indicator in use.	(*)
		BIT3	GALILEO Indicator in use.	(*)
		BIT4	Reference Indicator in use.	(*)
		BIT5	Single Receiver Oscillator Indicator in use.	(*)

(*) : BIT = 1 is In Use , BIT = 0 is not In use .

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'u'	STRING	-	-	1	1	ID Configure : POSITION_ANTENNA		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|181|W|u|000000000|60|60|60|60|60|60|60|1|1|47.082164|2.399216|10.000|GP|47.000000|2.000000|10.000|0|0|0|0|0|123|ADVNULLANTENNA|0.0.0.0|ESPRTK|3.9.0 _ 4.x.x|TS9PC_S9SUB|10000|20|20|1|96|*6575
ACK Respond	$ESP_OK|50|W|u|1|(ID : POSITION_ANTENNA) : Saved Success .|*3613
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

TIP ! : The table below describes the data fields to be used by the different types of messages when they are enabled.

Name	Standard	GGA	1005	1006	1007	1008	1032	1033	1004	1012
Rover Latitude(degree)	NMEA-0183	◉	○	○	○	○	○	○	○	○
Rover Longitude(degree)	NMEA-0183	◉	○	○	○	○	○	○	○	○
Rover Altitude(meter)	NMEA-0183	◉	○	○	○	○	○	○	○	○
Talker ID $__GGA	NMEA-0183	◉	○	○	○	○	○	○	○	○
Base Latitude (degree)(or auto convert to ECEF)	RTCM-DF026	○	◉	◉	○	○	◉	○	○	○
Base Longitude (degree)(or auto convert to ECEF)	RTCM-DF025	○	◉	◉	○	○	◉	○	○	○
Base Altitude (meter)(or auto convert to ECEF)	RTCM-DF027	○	◉	◉	○	○	◉	○	○	○

Base Antenna Height (meter)	RTCM-DF028	○	○	◉	○	○	○	○	○	○
Quarter Cycle Indicator	RTCM-DF364	○	◉	◉	○	○	○	○	○	○
Reserved for ITRF Realization Year	RTCM-DF021	○	◉	◉	○	○	◉	○	○	○
Reference Station ID	RTCM-DF003	○	◉	◉	◉	◉	◉	◉	○	○
Physical Reference Station ID (0,4095)	RTCM-DF226	○	○	○	○	○	◉	○	○	○
Antenna Setup ID	RTCM-DF031	○	○	○	◉	◉	○	◉	○	○
Antenna Descriptor	RTCM-DF030	○	○	○	◉	◉	○	◉	○	○
Antenna Serial Number	RTCM-DF033	○	○	○	○	◉	○	◉	○	○
Receiver Type Descriptor	RTCM-DF228	○	○	○	○	○	○	◉	○	○
Receiver Firmware Version	RTCM-DF230	○	○	○	○	○	○	◉	○	○
Receiver Serial Number	RTCM-DF232	○	○	○	○	○	○	◉	○	○
GPS Indicator GLONASS Indicator GALILEO Indicator Reference Indicator	RTCM-DF022 RTCM-DF023 RTCM-DF024 RTCM-DF141	○	◉	◉	○	○	○	○	○	○
Single Receiver Oscillator Indicator	RTCM-DF142	○	◉	◉	○	○	○	○	○	○
Max satellite GPS	ESPRTK	○	○	○	○	○	○	○	◉	◉
Max satellite GLONASS	ESPRTK	○	○	○	○	○	○	○	◉	◉
Max satellite SBAS	ESPRTK	○	○	○	○	○	○	○	◉	◉
Max OBS in an Epoch	ESPRTK	○	○	○	○	○	○	○	◉	◉

TIP ! : The table below describes the meanings of the fields defined according to the NMEA0183 and RTCM3.2 message standards.

Field	Data Field Notes
GGA - NMEA 0183	When used in the configuration field of ESPrtk, the input format is the geodetic coordinates (Lat-Long-Alt) used for NMEA-GGA messages.
RTCM3.2-DF025 RTCM3.2-DF026 RTCM3.2-DF027	The antenna reference point X-Y-Z-coordinate is referenced to ITRF epoch as given in DF021. When used in the configuration field of ESPrtk, the input format is geodetic coordinates (Lat-Long-Alt) and will be converted automatically to Cartesian ECEF (XYZ) coordinates when used in RTCM 1005 , 1006,1032 messages.
RTCM3.2-DF028	The Antenna Height field provides the height of the Antenna Reference Point above the marker used in the survey campaign.
RTCM3.2-DF364	The Quarter Cycle Indicator denotes whether different carrier phase signals tracked on the same frequency have a common phase, i.e. whether or not the fractional PhaseRanges of two signals on the same frequency show a quarter cycle difference (see also section 3.1.7 for further explanation).The definition of the indicator relates exclusively to the correction

	status of the quarter cycle, and applies to Messages Types 1001, 1002, 1003, 1004, 1009, 1010, 1011, 1012 . Other possible corrections cannot be indicated by this indicator. • 0 - Correction status unspecified • 1 - PhaseRanges in Message Types 1001, 1002, 1003, 1004, 1009, 1010, 1011, 1012 are corrected in such a way that whenever PhaseRanges for different signals on the same frequency are present in these messages, they are guaranteed to be in phase and thus shall show no Quarter-Cycle bias between them (see Table 3.1-5 for details on the adjustments made). Double differences of PhaseRanges tracked with different signals shall show no QuarterCycle differences. • 2 - Phase observations are not corrected. Double differences may show Quarter-Cycle differences for PhaseRanges based on different signals on the same frequency. Processing will require appropriate corrections. • 3 – Reserved
RTCM3.2-DF021	ITRF Realization Year . Since this field is reserved, all bits should be set to zero for now. However, since the value is subject to change in future versions, decoding should not rely on a zero value. The ITRF realization year identifies the datum definition used for coordinates in the message.
RTCM3.2-DF003	The Reference Station ID is determined by the service provider. Its primary purpose is to link all message data to their unique source. It is useful in distinguishing between desired and undesired data in cases where more than one service may be using the same data link frequency. It is also useful in accommodating multiple reference stations within a single data link transmission. In reference network applications the Reference Station ID plays an important role, because it is the link between the observation messages of a specific reference station and its auxiliary information contained in other messages for proper operation. Thus the Service Provider should ensure that the Reference Station ID is unique within the whole network, and that ID's should be reassigned only when absolutely necessary. Service Providers may need to coordinate their Reference Station ID assignments with other Service Providers in their region in order to avoid conflicts. This may be especially critical for equipment accessing multiple services, depending on their services and means of information distribution.
RTCM3.2-DF226	The Physical Reference Station ID specifies the station ID of a real reference station, when the data stream itself is based on a nonphysical reference station. Consequently, for the Physical Reference Station ID the same notes apply as for DF003.
RTCM3.2-DF031	• 0 - Use standard IGS Model . • 1-255 - Specific Antenna Setup ID# The Antenna Setup ID is a parameter for use by the service provider to indicate the particular reference station-antenna combination. The number should be increased whenever a change occurs at the station that affects the antenna phase center variations. While the Antenna Descriptor and the Antenna Serial Number give an indication of when the installed antenna has been changed, it is envisioned that other changes could occur. For instance the antenna may have been repaired, or the surrounding of the antenna may have been changed and the provider of the service may want to make the user station aware of the change. Depending on the change of the phase center variations due to a setup change, a change in the Antenna Setup ID would mean that the user should check with the service provider to see if the antenna phase center variation

	in use is still valid. Of course, the provider must make appropriate information available to the users.
RTCM3.2-DF030	Alphanumeric characters. IGS limits the number of characters to 20 at this time, but this DF allows more characters for future extension.
RTCM3.2-DF033	Alphanumeric characters. The Antenna Serial Number is the individual antenna serial number as issued by the manufacturer of the antenna. A possible duplication of the Antenna Serial Number is not possible, because together with the Antenna Descriptor only one antenna with the particular number will be available. In order to avoid confusion the Antenna Serial Number should be omitted when the record is used together with reverse reduction to model type calibration values, because it cannot be allocated to a real physical antenna.
RTCM3.2-DF228	Any text string for Receiver Type Descriptor can be use. Max length = 31 characters.
RTCM3.2-DF230	Receiver Firmware Version
RTCM3.2-DF232	Receiver Serial Number
RTCM3.2-DF022	• 0 - No GPS service supported • 1 - GPS service supported
RTCM3.2-DF023	• 0 - No GLONASS service supported • 1 - GLONASS service supported
RTCM3.2-DF024	• 0 - No Galileo service supported • 1 - Galileo service supported
RTCM3.2-DF141	Reference-Station Indicator . • 0 - Real, Physical Reference Station • 1 - Non-Physical or Computed Reference Station Note: A Non-Physical or Computed Reference Station is typically calculated based on information from a network of reference stations. Different approaches have been established over years. The NonPhysical or Computed Reference Stations are sometimes trademarked and may not be compatible. Examples of these names are “Virtual Reference Stations”, “Pseudo-Reference Stations”, and “Individualized Reference Stations”.
RTCM3.2-DF142	• 0 - All raw data observations in messages 1001-1004 and 1009-1012 may be measured at different instants. This indicator should be set to “0” unless all the conditions for “1” are clearly met. • 1 - All raw data observations in messages 1001-1004 and 1009-1012 are measured at the same instant, as described in Section 3.1.4.
Max satellite GPS	Max satellite GPS for RTCM-1004 generartor, range [1,32]. (Reducing the value will save the RAM for ESPrtk.)
Max satellite GLONASSS	Max satellite GLONASS for RTCM-1012 generartor, range [1,27]. (Reducing the value will save the RAM for ESPrtk.)
Max satellite SBAS	Max satellite SBAS, range [1,35]. (Reducing the value will save the RAM for ESPrtk.)

Max OBS in an Epoch	Max OBS in an Epoch, range [1,96]. (Reducing the value will save the RAM for ESPrtk.)

Command Write Configure - WIFI_TCP_UDP

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'v'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : WIFI_TCP_UDP
Number of field values : 9

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	5	0	-	-	Type Protocol	
PL3	PRINT	-	-	ESPRTK_WF	1	40	WiFi SSID / Hotspot's name	
PL4	PASS_A	-	-	12345678	0	40	WiFi Password	
PL5	STRING	-	-	192.168.4.2	1	40	IP Connect	
PL6	INTEGER	0	65535	4001	-	-	Port Connect	
PL7	STRING	-	-	192.168.4.1	1	40	Local IP	
PL8	INTEGER	0	65535	4000	-	-	Local Port	
PL9	STRING	-	-	192.168.4.0	1	40	Gateway	
PL10	STRING	-	-	255.255.255.0	1	40	Subnet	

PL0	PL1	PL2	Type Protocol	Unit
'W'	'e'	0	TCP Server Access Point	
		1	TCP Server Station	
		2	TCP Client Station	
		3	UDP Server Access Point	
		4	UDP Server Station	
		5	UDP Client Station	

Elements require	TCP Server AP	TCP Server STA	TCP Client STA	UDP Server AP	UDP Server STA	UPD Client STA
WIFI SSID (Name)	◉	◉	◉	◉	◉	◉
WiFi Password	◉	◉	◉	◉	◉	◉
IP Connect to	○	◉	◉	○	◉	◉
Port Connect to	○	○	◉	○	◉	◉
Local IP	◉	○	○	◉	○	○
Local Port	◉	◉	○	◉	◉	◉
Gateway IP	◉	○	○	◉	○	○
Subnet IP	◉	○	○	◉	○	○

Main Action select			Feature support					
			TCP Server AP	TCP Server STA	TCP Client STA	UDP Server AP	UDP Server STA	UPD Client STA
No Main Action UART_to_Bluetooth UART_Base_to_UART_Rover	< ignore ! >		◉	◉	◉	◉	◉	◉
MQTT_Base NTRIP_Base	Internet via WiFi		○	◉	◉	○	◉	◉
MQTT_Rover NTRIP_Rover	Internet via Ethernet/2G/3G/4G		◉	◉	◉	◉	◉	◉
MQTT_Rover_Repeater NTRIP_Rover_Repeater Radio_Rover_Repeater	Radio repeater via WiFi	Internet via WiFi / Ethernet / 2G/3G/4G	○	○	○	○	○	○
	Radio repeater via LoRa	Internet via WiFi	○	◉	◉	○	◉	◉
		Internet via Ethernet	○	○	○	○	○	○
		Internet via 2G/3G/4G	◉	◉	◉	◉	◉	◉
Radio_Base Radio_Rover UART_to_RF_Base UART_to_RF_Rover	Radio Wireless bridge via WiFi		○	○	○	○	○	○
Bluetooth_to_RF_Base Bluetooth_to_RF_Rover	Radio Wireless bridge via LoRa		◉	◉	◉	◉	◉	◉

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'v'	STRING	-	-	1	1	ID Configure : WIFI_TCP_UDP		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|85|W|v|2|ESPRTK_WF|12345678|192.168.1.3|4001|192.168.1.4|4000|192.168.4.1|255.255.255.0|*3635
ACK Respond	$ESP_OK|46|W|v|1|(ID : WIFI_TCP_UDP) : Saved Success .|*3C06
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - NTRIPCASTER_Admin

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'w'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : NTRIPCASTER_Admin
Number of field values : 9

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	STRING	-	-	admin	1	20	Name Login WebAdmin	
PL3	PASS_A	-	-	abc123ABC	1	20	Pass Login WebAdmin	
PL4	STRING	-	-	esprtk@gmail.com	1	60	Admin Email	
PL5	INTEGER	0	65535	80	-	-	WebAdmin Port	
PL6	INTEGER	0	65535	2101	-	-	Caster Port	
PL7	INTEGER	0	254	60	-	-	Cycle Update Parse NMEA	second
PL8	INTEGER	0	254	9	-	-	Basestation timeout data receive	second
PL9	INTEGER	3600	20000000	86400	-	-	Cycle save record limit	second
PL10	INTEGER	0	1	1	-	-	Enable Print Status to TX_UART_0	

PL0	PL1	PL7	Cycle Update Parse NMEA	Unit
'W'	'w'	0	Disable Update Parse NMEA	
		>= 1	Enable Update Parse NMEA	

PL0	PL1	PL8	Basestation timeout data receive	Unit
'W'	'w'	0	Disable Basestation timeout data receive	
		>= 1	Enable Basestation timeout data receive	

PL0	PL1	PL9	Cycle save record limit	Unit
'W'	'w'	0	Disable save record limit	
		1	Enable save record limit	second

PL0	PL1	PL10	Enable Print Status to TX_UART_0	Unit
'W'	'w'	0	Disable Print Status to TX_UART_0	
		1	Enable Print Status to TX_UART_0	

Payload Index	Describe	Detail	Unit
PL2	Name Login WebAdmin	For Admins/Owners of NTRIP CASTER, used to login to Dashboard site to manage user accounts.	
PL3	Pass Login WebAdmin		
PL4	Admin Email	When NTRIP CASTER detected client error or abuse. CASTER will send an error message with the email address of the administrator (admin) via NTRIP Stream , so customers can contact them to resolve the issue.	
PL5	WebAdmin Port	The port that NTRIP CASTER uses to run the Dashboard page. Usually 80. Then , the Dashboard's address will be **http://x.x.x.x/admin** or **http://x.x.x.x:80/admin** (With **x.x.x.x** is the local IP address of that NTRIP CASTER) . Setting this value to 0 will not activate the Dashboard and increase the number of sockets by 1 for the NTRIP Client / Server.	
PL6	Caster Port	The port that NTRIP CASTER uses to listen to and respond to connections from the NTRIP Client (Rover / Base). Usually 2101. Then , the Sourcetable Page's address will be **http://x.x.x.x:2101** or **http://x.x.x.x:80** (With **x.x.x.x** is the local IP address of that NTRIP CASTER)	
PL7	Cycle Update Parse NMEA	NTRIP CASTER will collect the Rover's location via GGA / NMEA message. Setting the	second

		cycle as small will help show the location with a lower delay. Recommended value is above 60s. Setting this value to 0 disables this function.	
PL8	Basestation timeout data receive	NTRIP CASTER acts as a data bridge, it receives RTCM data from NTRIP Base and distributes it to all Rovers. If NTRIP Base stops sending RTCM data after a period of time, NTRIP CASTER will understand the NTRIP Base has lost connection (died or stopped working). The default value is 9 seconds.	second
PL9	Cycle save record limit	All changes, user configuration data, bandwidth counter, etc. will be automatically saved to FLASH periodically after a period of time. The default value is 86400 seconds. (1 day). Setting this value to 0 disables this function.	second
PL10	Enable Print Status to TX_UART_0		

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'w'	STRING	-	-	1	1	ID Configure : NTRIPCASTER_Admin		
	PL3	INTEGER	0	1	-	-	Status Configure		
							0	Error: Failed to save , please try again	
							1	OK : Saved configure success	
	PL4	STRING	-	-	-	-	Describe Status Configure in text string format		

Example

Message :	
HOST Send	$ESP_OK|58|W|w|admin|abc123ABC|esprtk@gmail.com|80|2101|60|9|86400|1|*675B
ACK Respond	$ESP_OK|51|W|w|1|(ID : NTRIPCASTER_Admin) : Saved Success .|*7F40
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - NTRIPCASTER_ACC

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'x'	Infor 1	Infor 2	Infor 3	Infor 4	...

Name CFG : NTRIPCASTER_ACC
Number of field values : 180

Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL2	STRING	-	-	name_0	1	10	Login Name	
PL3	PASS_A	-	-	pass_0	1	10	Password	
PL4	INTEGER	0	10	10	-	-	Number Rover	
PL5	STRING	-	-	Base_0	1	10	Mountpoint	
PL6	INTEGER	0	1	0	-	-	VRS Support	
PL7	FLOAT	0.000	1000000000	0	-	-	Limit Highest Speed Bandwidth	bps
PL8	INTEGER	0	10000000	0	-	-	Limit Total Bandwidth (Byte Count)	GByte
PL9	INTEGER	0	2000000000	0	-	-	Limit time running	second
PL10	FLOAT	0.000	1000000000	0	-	-	Offset Highest Speed Bandwidth (counter start from this value)	bps
PL11	INTEGER	0	10000000	0	-	-	Offset Total Bandwidth (Byte Count) . (counter start from this value)	GByte
PL12	INTEGER	0	2000000000	0	-	-	Offset time running (counter start from this value)	second
PL13	INTEGER	0	1	0	-	-	Enable use STR from Client/User	
PL14	PRINT	-	-	Tokyo_JAPAN;RTCM 3.2;1005(1),1077(1),1087(1);3;GPS+GLO;S	0	250	STR String . (<identifier><format><format-details><carrier><nav-system><network><country><latitude><longitude><nmea><solu	

				NIP;JPN;35.68;139.69;0;1;sNTRIP;yes;B;N;3620;none			tion><generator><compression><authentication><fee><bitrate><misc>)	
PL15 to PL27	Configure field from PL15 to PL27 (for Account #d) is the same with Configure feild from PL2 to PL14							
PL28 to PL40	Configure field from PL28 to PL40 (for Account #d) is the same with Configure feild from PL2 to PL14							
PL41 to PL53	Configure field from PL41 to PL53 (for Account #d) is the same with Configure feild from PL2 to PL14							
PL54 to PL66	Configure field from PL54 to PL66 (for Account #d) is the same with Configure feild from PL2 to PL14							
PL67 to PL79	Configure field from PL67 to PL79 (for Account #d) is the same with Configure feild from PL2 to PL14							
PL80 to PL92	Configure field from PL80 to PL92 (for Account #d) is the same with Configure feild from PL2 to PL14							
PL93 to PL105	Configure field from PL93 to PL105 (for Account #d) is the same with Configure feild from PL2 to PL14							
PL93 to PL105	Configure field from PL106 to PL118 (for Account #d) is the same with Configure feild from PL2 to PL14							
PL119 to PL131	Configure field from PL119 to PL131 (for Account #d) is the same with Configure feild from PL2 to PL14							
Payload Index	Type value	Min Value	Max Value	Default Value	Min Length	Max Length	Describe	Unit
PL132	HEX_Up	00000000	FFFFFFFF	00000000	-	-	Ban IP (IP #1)	
PL133 to PL182	Configure field from PL133 to PL182 (IP #2 → IP #50) is the same with Configure feild PL132							

ESPrtk Respond

Payload	Type value	Min	Max	Min	Max	Describe	Unit

<table>
<tr><th colspan="2">Index</th><th></th><th>Value</th><th>Value</th><th>Length</th><th>Length</th><th colspan="2"></th><th></th></tr>
<tr><td>PL2</td><td>‘x’</td><td>STRING</td><td>-</td><td>-</td><td>1</td><td>1</td><td colspan="2">ID Configure : NTRIPCASTER_ACC</td><td></td></tr>
<tr><td rowspan="3"></td><td rowspan="3">PL3</td><td rowspan="3">INTEGER</td><td rowspan="3">0</td><td rowspan="3">1</td><td rowspan="3">-</td><td rowspan="3">-</td><td colspan="2">Status Configure</td><td rowspan="3"></td></tr>
<tr><td>0</td><td>Error: Failed to save , please try again</td></tr>
<tr><td>1</td><td>OK : Saved configure success</td></tr>
<tr><td></td><td>PL4</td><td>STRING</td><td>-</td><td>-</td><td>-</td><td>-</td><td colspan="2">Describe Status Configure in text string format</td><td></td></tr>
</table>

Example

<table>
<tr><th colspan="2">Message :</th></tr>
<tr><td>HOST Send</td><td>$ESP_OK|587|W|x|name_0|pass_0|10|Base_0|0|0|0|0|0|0|0|0|Tokyo_JAPAN;RTCM 3.2;1005(1),1077(1),1087(1);3;GPS+GLO;SNIP;JPN;35.68;139.69;0;1;sNTRIP;yes;B;N;3620;none|name_1|pass_1|10|Base_1|0|0|0|0|0|0|0|0|REZONVILLE_FRA;RTCM 3.2;1005(1),1077(1),1087(1),1097(1),1127(1);3;GPS+GLO+GAL+BDS;SNIP;FRA;49.10;5.98;0;1;sNTRIP;yes;B;N;7620;none|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|00000000||||||||||||||||||||||||||||||||||||||||||||||||||*4C0F</td></tr>
<tr><td>ACK Respond</td><td>$ESP_OK|49|W|x|1|(ID : NTRIPCASTER_ACC) : Saved Success .|*7709</td></tr>
<tr><td>NACK Respond</td><td>$ESP_ER|… or $ESS_ER|… or $AES_ER|…</td></tr>
</table>

Start UART Configure mode for Read configure data.

Use ESPrtk command to start UART configure

Use ESPrtk command control	
Step 0	ESPrtk after supplying power.
Step 2	HOST send command : **$ESP_OK|5|O|_||*441E**
Done	ESPrtk will start go to WEB Configure mode. The CONNECT_LED blink fast.
Success	The CONNECT_LED will light continuously . ESPrtk respond on TX_UART0: **$ESP_OK|5|O|_||*441E**

Command Read Configure ESPrtk

Tip : To use these command , ESPrtk need running at UART Configure mode !

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'R'	ID Command	Infor 1	Infor 2	Infor 3	Infor 4	...

PL0	PL1	Describe Type ID Command	Example
'R'	'a'	CREATE NEW PROFILE	
'R'	'b'	LOGIN_LOGOUT	
'R'	'c'	Read YOUR_PROFILE 's configure data	$ESP_OK|5|R|c||*653F
'R'	'd'	Read ACTION_PLANNING 's configure data	$ESP_OK|5|R|d||*6238
'R'	'e'	Read INTNERNET 's configure data	$ESP_OK|5|R|e||*6339
'R'	'f'	Read RADIO_RTK 's configure data	$ESP_OK|5|R|f||*603A
'R'	'g'	Read MQTT_RTK 's configure data	$ESP_OK|5|R|g||*613B
'R'	'h'	Read NTRIP_CLIENT 's configure data	$ESP_OK|5|R|h||*6E34

‘R’	‘i’	Read NTRIP_MASTER ‘s configure data	$ESP_OK|5|R|i||*6F35
‘R’	‘j’	Read UART_CONFIGURE ‘s configure data	$ESP_OK|5|R|j||*6C36
‘R’	‘k’	Read BLUETOOTH ‘s configure data	$ESP_OK|5|R|k||*6D37
‘R’	‘l’	Read DISPLAY_VIEWER ‘s configure data	$ESP_OK|5|R|l||*6A30
‘R’	‘m’	Read EVENT_LOG ‘s configure data	$ESP_OK|5|R|m||*6B31
‘W’	‘n’	Request convert ID Error to text - SIMPLE_DEBUG	$ESP_OK|7|W|n|23|*6E07
‘R’	‘o’	Read SD_CARD ‘s configure data	$ESP_OK|5|R|o||*6933
‘R’	‘p’	Read IMU_MPU9250 ‘s configure data	$ESP_OK|5|R|p||*762C
‘R’	‘q’	Read GPIO_PIN_MAP ‘s configure data	$ESP_OK|5|R|q||*772D
‘R’	‘r’	Read SYSTEM_Other_Mode ‘s configure data	$ESP_OK|5|R|r||*742E
‘R’	‘s’	Read GSM_CELLULAR ‘s configure data	$ESP_OK|5|R|s||*752F
‘R’	‘t’	Read STREAM_MANAGER‘s configure data	$ESP_OK|5|R|t||*7228
‘R’	‘u’	Read POSITION_ANTENNA‘s configure data	$ESP_OK|5|R|u||*7329
‘R’	‘v’	Read WIFI_TCP_UDP‘s configure data	$ESP_OK|5|R|v||*702A
‘R’	‘w’	Read NTRIPCASTER_Admin‘s configure data	$ESP_OK|5|R|w||*712B
‘R’	‘x’	Read NTRIPCASTER_ACC‘s configure data	$ESP_OK|5|R|x||*7E24
‘R’	‘y’	Request ERROR_SYSTEM	$ESP_OK|5|R|y||*7F25
‘R’	‘z’	Request VERSION_MAC ‘s configure data	$ESP_OK|5|R|z||*7C26

Command Read configure - YOUR_PROFILE

Note: This configure only work on WEB Configure interface !

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|c||*653F
ACK Respond	$ESP_OK|80|R|c|ESPrtk : IHGG4|def123DEF|0|admin|abc123ABC|ESPrtk new user|Hello member|1|0|*164C
NACK Respond	$ESP_ER|76|R|j|NACK : Only support this action on WEB Configure or Command Tester Tool|*6F38

Command Read configure - ACTION_PLANNING

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|d||*6238
ACK Respond	$ESP_OK|41|R|d|10|1|8|6|6|1|0|0|0000000000000000000|*5B5C
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - INTNERNET

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|e||*6339
ACK Respond	$ESP_OK|67|R|e|0|1|TEWM_B31B6B|884744F797|Tran Trong Toan|12121975|My Router||*1C0E
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - RADIO_RTK

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|f||*603A
ACK Respond	$ESP_OK|52|R|f|abc_LoRa_WiFi_012345|0|0|9|0|433000000|0|9|8|10|*1C06
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - MQTT_RTK

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|g||*613B
ACK Respond	$ESP_OK|104|R|g|m11.cloudmqtt.com|18052|rkdpcppo|nvZv101KE_SX|9R082_O68R1|RTCM|NMEA|FAST_NMEA|1|abc_MQTT_RTK_012345|*3855
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - NTRIP_CLIENT

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|h||*6E34
ACK Respond	$ESP_OK|31|R|h|rtk2go.com|2101|no|no|no|1|*0F7D
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - NTRIP_MASTER

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|i||*6F35
ACK Respond	$ESP_OK|173|R|i|rtk2go.com|2101|A00__81909_GHB9A|BETATEST2|1|0|San_Jose-Calif|13|1005(1),1077(1),1107(1),1117(1)|1|1|1|1|0|1|0|1|0|0|0|SNIP|USA|37.388|-121.9|1|0|0|1|0|8600|ESPrtk-Base|*491E
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - UART_CONFIGURE

Note: This configure only work on WEB Configure interface !

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|j||*6C36
ACK Respond	$ESP_OK|48|R|j|1|1|0|0|1|2|3|4|5|6|7|8|9|10|11|12|13|14|15|*6508
NACK Respond	$ESP_ER|76|R|j|NACK : Only support this action on WEB Configure or Command Tester Tool|*6F38

Command Read configure - BLUETOOTH

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|k||*6D37

ACK Respond	$ESP_OK|56|R|k|0|BTU-ESPrtk : 81909_GHB9A|BLE-ESPrtk : 81909_GHB9A|*6332
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - DISPLAY_VIEWER

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|l||*6A30
ACK Respond	$ESP_OK|20|R|l|1|6|6|0|1|0|1|0|*2009
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - EVENT_LOG

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|m||*6B31
ACK Respond	$ESP_OK|14|R|m|3|0|0|0|0|*6876
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Request convert ID Error to text - SIMPLE_DEBUG

HOST send request convert ID Error to text

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'W'	'n'	ID_Error	Infor 1	Infor 2	Infor 3	...

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	255	-	-	ID_ERROR	

ESPrtk Respond

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	INTEGER	0	255	-	-	ID_ERROR	
PL3	STRING	-	-	-	-	Type of error in string	
PL4	STRING	-	-	-	-	Detail describe error in string.	
PL5	STRING	-	-	-	-	Guess the cause of this error and Suggestions for how to resolve this error.	

Example

Message :	
HOST Send	$ESP_OK|7|W|n|39|*6506
ACK Respond	$ESP_OK|130|W|n|39|SETTUP|No any Actions to do !| ,How_to_fix: Go to Web Configure -> tab Action Planning -> Enable action for ESPrtk |*0C52
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

	Describe
ID_ERROR	39
Type	SETTUP
Detail	No any Actions to do !
Suggestions	 ,How_to_fix: Go to Web Configure -> tab Action Planning -> Enable action for ESPrtk

TIP :
 mean Enter / Page break

Command Read configure - SD_CARD

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|o||*6933
ACK Respond	$ESP_OK|59|R|o|my_record|0|0|500|1|2|11000000000000000000000000000000|*2005
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command read configure - IMU_MPU9250

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|p||*762C
ACK Respond	$ESP_OK|61|R|p|1|3|1|0|4.512|000000111|00001111|0|0|0|1|0|0|0|1|0|0|0|1|*5B49
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - GPIO_PIN_MAP

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|q||*772D
ACK Respond	$ESP_OK|72|R|q|18|4|21|1|16|10|14|0|17|11|13|7|8|2|22|19|12|9|23|20|15|3|24|6|25|5|*3A0B
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - SYSTEM_Other_Mode

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|r||*742E
ACK Respond	$ESP_OK|50|R|r|9|3|3|3|3|3|111111111111|11111111111111111111|*3272
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command read configure - GSM_CELLULAR

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|s||*752F
ACK Respond	$ESP_OK|249|W|s|0|0|1234|w3-world|mms|mms|0|7|0|10000000000000000000| 10000000000000000000000000000000000000000000000000000000000000000000000000000000|10000000|[RST] [COM^9600][DLY^1][ATE0^5^10^OK][AT+CMEE=0^5^10^OK^ok][CLR][PIN^1234][CFG] [NET^w3-world^mms^mms][COM^115200]|*0F54
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - STREAM_MANAGER

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|t||*7228
ACK Respond	$ESP_OK|17|W|t|0|0|0|0|0|0||*444E
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - POSITION_ANTENNA

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|u||*7329
ACK Respond	$ESP_OK|181|R|u|000000000|60|60|60|60|60|60|60|1|1|47.082164|2.399216|10.000|GP|47.000000|2.000000|10.000|0|0|0|0|0|123|ADVNULLANTENNA|0.0.0.0|ESPRTK|3.9.0 _ 4.x.x|TS9PC_S9SUB|10000|20|20|1|96|*6070
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - WIFI_TCP_UDP

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|v||*702A
ACK Respond	$ESP_OK|85|R|v|2|ESPRTK_WF|12345678|192.168.1.3|4001|192.168.1.4|4000|192.168.4.1|255.255.255.0|*3330
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - NTRIPCASTER_Admin

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|w||*712B
ACK Respond	$ESP_OK|58|R|w|admin|abc123ABC|esprtk@gmail.com|80|2101|60|9|86400|1|*625B
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Read configure - NTRIPCASTER_ACC

ESPrtk Respond

The index / range of values / data types is similar to the index / range of values / data types of payloads when configured.

Example

Message :	
HOST Send	$ESP_OK|5|R|x||*7E24
ACK Respond	$ESP_OK|622|R|x|name_0|pass_0|10|Base_0|0|0.000000000|0|0|0.000000|0|5413|0|Tokyo_JAPAN;RTCM3.2;1005(1),1077(1),1087(1);3;GPS+GLO;SNIP;JPN;35.68;139.69;0;1;sNTRIP;yes;B;N;3620;none|name_1|pass_1|10|Base_1|0|0.000000000|0|0|0.000000|0|0|1|REZONVILLE_FRA;RTCM3.2;1005(1),1077(1),1087(1),1097(1),1127(1);3;GPS+GLO+GAL+BDS;SNIP;FRA;49.10;5.98;0;1;sNTRIP;yes;B;N;7620;none|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|_|_|0|_|0|0|0|0|0|0|0|0|0|00000000|||||||||||||||||||||||||||||||||||||||||||||||||*7705
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Request ERROR_SYSTEM

ESPrtk Respond

PL	Name type error	PL	Name type error
PL2	FIRST SETTING	PL12	DISPLAY_VIEWER
PL3	UART	PL13	UART_CONIGURE
PL4	INTERNET	PL14	EVENT_LOG
PL5	MQTT	PL15	Other1
PL6	NTRIP	PL16	Other2
PL7	RADIO_RF_PHY	PL17	Other3
PL8	RADIO_RF_ROVER	PL18	Other4
PL9	BLUETOOTH	PL19	Other5
PL10	SD_CARD	PL20	Other6
PL11	IMU_MPU	PL21	Other7

Example

Message :	
HOST Send	$ESP_OK|5|R|y||*7F25
ACK Respond	$ESP_OK|44|R|y|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|*360F
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Request VERSION_MAC ‘s configure data

ESPrtk Respond

Payload Index	Type value	Format	Example	Describe	Unit

PL2	PRINT	%d/%d/%d	1/1/20	Date/ Month /Year ESPrtk Version	
PL3	PRINT	%d.%d.%d	3.0.0	ESPrtk Version	
PL4	PRINT	%X02:%X02:%X02:%X02:%X02:%X02	30:AE:A4:28:24:90	6 bytes MAC Address	
PL5	PRINT	%d	1	CHIP Revision	
PL6	PRINT	%d	2	Number of cores of ESP32	

Example

Message :	
HOST Send	$ESP_OK|5|R|z||*7C26
ACK Respond	$ESP_OK|41|R|z|21/12/19|2.7.9|30:AE:A4:28:24:90|1|2|*3627
NACK Respond	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Start UART Configure mode for Hardware control.

Use ESPrtk command to start UART configure

Use ESPrtk command control	
Step 0	ESPrtk after supplying power.
Step 2	HOST send command : **$ESP_OK|5|O|_||*441E**
Done	ESPrtk will start go to WEB Configure mode. The CONNECT_LED blink fast.
Success	The CONNECT_LED will light continuously . ESPrtk respond on TX_UART0: **$ESP_OK|5|O|_||*441E**

Command Hardware control ESPrtk

Tip : To use these command , ESPrtk need running at UART Configure mode !

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	..	…
$ESP_OK|xx|	‘T’	ID Command	ID Control	Infor 1	Infor 2	...	...

PL0	PL1		**Describe Type Command**	**Numbers of ID Control**
‘T’	‘P'	PL2	**Control Enable Type Print on TX_UART0**	1
		‘0’	P_Control Enable Type Print on TX_UART0	
‘T’	‘G’	PL2	**Control GPIO on ESPrtk**	3
		‘0’	G_SET_MODE_GPIO	
		‘1’	G_WRITE_DIGITAL	
		‘2’	G_READ_DIGITAL	

'T'	'U'	PL2	**Control UART_1 and UART_2 on ESPrtk**	8
		'0'	U_Start_UART	
		'1'	U_Check_UART_Started	
		'2'	U_Send_String	
		'3'	U_Send_Binary	
		'4'	U_Available	
		'5'	U_Read_String	
		'6'	U_Read_Binary	
		'7'	U_Clear_Buffer	
'T'	'I'	PL2	**Control WIFI and Ethernet module connect to router**	6
		'0'	I_Start_WIFI_and_Connect_to_Router	
		'1'	I_WIFI_Disconnect_to_Router	
		'2'	I_Start_Ethernet_ENC28j60_and_Connect_to_Router	
		'3'	I_Ethernet_ENC28j60_Disconnect_to_Router	
		'4'	I_Start_Ethernet_W5XX_and_Connect_to_Router	
		'5'	I_Ethernet_W5XX_Disconnect_to_Router	
'T'	'T'	**PL2**	Control TCP Client Task communicate with Server	9
		'0'	T_Connect_to_Server	
		'1'	T_Check_Connected_to_Server	
		'2'	T_Send_String	
		'3'	T_Send_Binary	
		'4'	T_Available	
		'5'	T_Read_String	
		'6'	T_Read_Binary	

		'7'	T_Clear_Buffer	
		'8'	T_Diconnect_to_Server	
'T'	'B'	**PL2**	**Control Bluetooth Classic and Bluetooth BLE on ESPrtk**	8
		'0'	B_Start_Bluetooth	
		'1'	B_Check_Client_Connected	
		'2'	B_Send_String	
		'3'	B_Send_Binary	
		'4'	B_Available	
		'5'	B_Read_String	
		'6'	B_Read_Binary	
		'7'	B_Clear_Buffer	
'T'	'O'	**PL2**	**Control OLED Display (SSD1306/SH1106)**	12
		'0'	O_Start_OLED	
		'1'	O_Update_and_Display	
		'2'	O_Clear	
		'3'	O_Draw_Dot	
		'4'	O_Draw_Line	
		'5'	O_Draw_Rect	
		'6'	O_Draw_Rect_Corner	
		'7'	O_Draw_Triangle	
		'8'	O_Draw_Circle	
		'9'	O_Draw_Elippse	
		'A'	O_Draw_Character	
		'B'	O_Draw_Number	

'T'	'S'	**PL2**	**Control SD Card**	10
		'0'	S_Start_SD_Card	
		'1'	S_Check_Folder_Exist	
		'2'	S_Create_Folder	
		'3'	S_Delete_File	
		'4'	S_Rename_File	
		'5'	S_Check_File_Exist	
		'6'	S_Create_File	
		'7'	S_Get_Size_File	
		'8'	S_Read_Binary_data_File	
		'9'	S_Append_Binary_data_to_File	
'T'	'M'	**PL2**	**Control IMU (MPU9250)**	5
		'0'	M_Start_Print_IMU_Data	
		'1'	M_Stop_Print_IMU_Data	
		'2'	M_Start_IMU_Calibrate	
		'3'	M_Stop_IMU_Calibrate	
		'4'	M_Save_Calibrate_Result	
			Disable/Enable type IMU data	
'T'	'V'	**PL2**	**Control Deep Sleep**	2
		'0'	V_Enter_Deep_Sleep_mode	
		'1'	V_Exit_Deep_Sleep_mode_and_WakeUp	

Command Hardware Control – Control Enable Type Print on TX_UART0

Tip : This control command woks on UART configure mode , Main Action mode and WEB configure mode via UART0.

HOST send Control Enable Type Print on TX_UART0

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'P'	ID Control	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'0'						ID Control : Enable Type Print on TX_UART0	
	PL3	BIT	-	-	12	12	BIT Enable Type Print on TX_UART0	

PL0	PL1	PL2	PL3	Enable print message types on TX_UART0 port	Unit
'T'	'P'	'0'	BIT1	_SYSTEM_ (ESPrtk System)	
			BIT2	_ERROR_ (Error status)	
			BIT3	_INFO_ (Text , Log, Information...)	
			BIT4	_GOIN_ (Go to function status)	
			BIT5	_WFSCAN_ (Wifi Scan Result)	
			BIT6	_BROKER_ (Broker MQTT Status)|	
			BIT7	_RVNMEA_ (print NMEA from Rover)|	
			BIT8	_CONFIG_ (Configure information detail)	
			BIT9	$__GGA (Rover print GGA message)	
			BIT10	$_NMEA_ (Rover print NMEA message)	
			BIT11	$ESPIM1 (IMU MPU9250 Data type 1)	
			BIT12	$ESPIM2 (IMU MPU9250 Data type 2)	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'0'						ID Control : Enable Type Print on TX_UART0		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: BIT enable PRINT invalid	
							0	-	
							1	OK	
	PL4	BIT	-	-	12	12	BIT Enable Type Print on TX_UART0		

Example

Message :	
HOST Send	$ESP_OK|19|T|P|0|111111111111|*2157
ACK Respond	$ESP_OK|21|T|P|0|1|111111111111|*676E
NACK Respond	$ESP_ER|40|T|P|0|-1|NACK: BIT enable PRINT invalid|*1D14
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Hardware Control – Control GPIO on ESPrtk

Tip : This control command woks on UART configure mode , Main Action mode and WEB configure mode via UART0.

‘T’	‘G’	PL2	Control GPIO on ESPrtk	
		‘0’	G_SET_MODE_GPIO	
		‘1’	G_WRITE_DIGITAL	
		‘2’	G_READ_DIGITAL	

HOST send Control G_SET_MODE_GPIO

Not support !

HOST send Control G_READ_DIGITAL

Not support !

HOST send Control G_WRITE_DIGITAL

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘G’	ID Control	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘1’						ID Control : G_WRITE_DIGITAL		
	PL3	BIT	-	-	19	19	BIT control GPIO ESPrtk		
							0	Set GPIO to LOW Level	
							1	Set GPIO to HIGH Level	

PL0	PL1	PL2	PL3	GPIO ESPrtk	PL3	GPIO ESPrtk

'T'	'P'	'0'	BIT1	2	BIT11	19
			BIT2	4	BIT12	21
			BIT3	5	BIT13	22
			BIT4	12	BIT14	23
			BIT5	13	BIT15	25
			BIT6	14	BIT16	26
			BIT7	15	BIT17	27
			BIT8	16	BIT18	32
			BIT9	17	BIT19	33
			BIT10	18		

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'1'						ID Control : G_WRITE_DIGITAL		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: BIT control GPIO invalid	
							0	-	
							1	OK	
	PL4	BIT	-	-	19	19	BIT control GPIO		
							0	GPIO now is LOW Level	
							1	GPIO now is HIGH Level	
							X	This GPIO was used by other task	

Example

Message :	
HOST Send	$ESP_OK|26|T|G|1|1111111111111111|*0A38
ACK Respond	$ESP_OK|28|T|G|1|1|1XXXX1XXX111111XX11|*2044
NACK Respond	$ESP_ER|40|T|G|1|-1|NACK: BIT control GPIO invalid|*2D5D
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Tip 1 : When BIT respond is 'X' , that mean the GPIO pin was used by other task (such as UART_1_2 , LoRa , IMU, Ethernet , SD_Card, NeoPixel, OLED, ERROR_LED , CONNECT_LED ….etc).

For example, if you use the pin map ESPrtk as default configure like this :

Configure	JSON Configure
Default from factory	{ "CWB_Button" : "13" , "RX_UART0" : "3" , "MOSI_SD_Card" : "23" , "MOSI_LoRa_Ethe" : "33" , "TX_UART0" : "1" , "MISO_SD_Card" : "39" , "MISO_LoRa_Ethe" : "35" , "RX_UART1" : "17" , "SCK_SD_Card" : "18" , "SCK_LoRa_Ethe" : "25" , "TX_UART1" : "5" , "CONNECT_LED" : "15" , "CS_SD_Card" : "19" , "RX_UART2" : "16" , "ERROR_LED" : "12" , "CS_LoRa" : "32" , "TX_UART2" : "4" , "TX_UART3" : "14" , "CS_Ethernet__PWR_GSM" : "0" , "SCL_I2C" : "22" , "SCL_OLED" : "27" , "RST_LoRa_Ethe" : "2" , "SDA_I2C" : "21" , "SDA_OLED" : "26" , "DI0_LoRa" : "34" , "RX_UART3" : "36" }

To use GPIO for Hardware Control , you must release them before.

External deives	To Release	Release
SD_Card	At Write Configure – ACTION_PLANNING (at configure Other Action) → Disable all BIT save data to SD_Card	**23 , 39 , 18 , 19**
LoRa Ethernet	At Write Configure – RADIO_RTK → Select Type radio link is WIFI to release LoRa	**33 , 35 , 25 , 2 , 34 , 0 , 32**
	At Write Configure – INTNERNET → Select Type internet is WIFI only to release Ethernet	
Sensor/IMU	At Write Configure – ACTION_PLANNING (at configure Other Action) → Disable BIT export IMU data	**22 , 21**
CONNECT_LED	At Write Configure – DISPLAY_VIEWER → Disable use CONNECT_LED	**15**
ERROR_LED	At Write Configure – DISPLAY_VIEWER → Disable use ERROR_LED	**12**

Oled	At Write Configure – DISPLAY_VIEWER → Select Type Display is NEOPIXEL	**27 , 26**
TX_UART12	At Write Configure – ACTION_PLANNING → Select Map TX-UART-RTK to other TX UART port	**5 (or 4)**
RX_UART12	At Write Configure – ACTION_PLANNING → Select Map RX-UART-RTK to other RX UART port	**17 (or 16)**

Tip 2 : GPIO pins in the release state (idle / not use / free) can be used for **GPIO control** . However, when users use other control functions such as Ethernet / IMU / SD Card, etc., IO pins will be taken back control for those functions !. (GPIO control will ignore those pins if they are used again.)

Command Write Configure - Control UART_1 and UART_2 on ESPrtk

'T'	'U'	PL2	**Control UART_1 and UART_2 on ESPrtk**	8
		'0'	U_Start_UART	
		'1'	U_Check_UART_Started	
		'2'	U_Send_String	
		'3'	U_Send_Binary	
		'4'	U_Available	
		'5'	U_Read_String	
		'6'	U_Read_Binary	
		'7'	U_Clear_Buffer	

HOST send Control U_Start_UART

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'U'	'0'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'0'						ID Control : U_Check_UART_Started	
	PL3	INTEGER	1	2	-	-	Select UART1 or UART2 for UART_CF	

	PL4	INTEGER	2400	1000000	-	-	Baudrate UART_CF	bps
	PL5	INTEGER	255	2000	-	-	Buffer size RX of UART_CF	byte

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'0'						ID Control : U_Check_UART_Started		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	-	
							1	OK	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|19|T|U|0|1|57600|1000|*2051
ACK Respond	$ESP_OK|61|T|U|0|1|Start UART_1 success,Baudrate = 57600,RX_Size = 1000|*4C2E
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control U_Check_UART_Started

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'U'	'1'	Infor 1	Infor 2	Infor 3	...

Payload Index	Type value	Min Valu	Max Value	Min Length	Max Length	Describe	Unit

			e					
PL2	‘1’	STRING	-	-	1	1	ID Control : U_Check_UART_Started	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘1’						ID Control : U_Check_UART_Started		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	UART not started	
							1	UART started	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|6|T|U|1|*6726
ACK Respond	$ESP_OK|31|T|U|1|1|UART Connected success|*4C0F
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control U_Send_String

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘U’	‘2’	Infor 1	Infor 2	Infor 3	...

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit

PL2	'2'						ID Control : U_Send_String	
	PL3	STRING	-	-	1	1500	Data send in string format	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'2'						ID Control : U_Send_String		
	PL3	INTEGER	-1	2	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : UART_CF was not started .	
							1	Error: Busy TX is sending	
							2	OK: Send String success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|29|T|U|2|My String out UART\r\n|*645B
ACK Respond	$ESP_OK|33|T|U|2|2|UART Send String success|*3844
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control U_Send_Binary

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'U'	'3'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'3'						ID Control : U_Send_Binary	
	PL3	HEX	-	-	2	1400	Data send in string format	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'3'						ID Control : U_Send_Binary		
	PL3	INTEGER	-1	2	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : UART_CF was not started .	
							1	Error: Busy TX is sending	
							2	OK: Send Binary success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|17|T|U|3|A0B0D0E12F|*5E6F
ACK Respond	$ESP_OK|33|T|U|3|2|UART Send Binary success|*2354
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control U_Available

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…

$ESP_OK|xx|	'T'	'U'	'4'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'4'	STRING	-	-	1	1	ID Control : U_Available	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'4'						ID Control : U_Available		
	PL3	INTEGER	-1	1000	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							[0,1000]	Size available RX buffer UART	

Example

Message :	
HOST Send	$ESP_OK|6|T|U|4|*6226
ACK Respond	$ESP_OK|11|T|U|4|1000|*295F
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control U_Read_String

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'U'	'5'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'5'						ID Control : U_Read_String	
	PL3	INTEGER	1	1000	-	-	Size read buffer UART	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'5'						ID Control : U_Read_String		
	PL3	INTEGER	-1	1000	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							[0,1000]	Size of string respond	
	PL4	STRING	-	-	0	1000	Data read buffer in String format		

Example

Message :	
HOST Send	$ESP_OK|9|T|U|5|12|*1307
ACK Respond	$ESP_OK\|22\|T\|U\|5\|12\|$GPGGA,06350|*3827
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control U_Read_Binary

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'U'	'6'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'6'						ID Control : U_Read_Binary	
	PL3	INTEGER	1	1000	-	-	Size read buffer UART	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'6'						ID Control : U_Read_Binary		
	PL3	INTEGER	-1	1000	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							[0,1000]	Size of string respond	
	PL4	STRING	-	-	-	-	Data read buffer in HEX format		

Example

Message :	
HOST Send	$ESP_OK|9|T|U|6|12|*1004
ACK Respond	$ESP_OK|34|T|U|6|12|332E3030302C323033342E36|*203B
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control U_Clear_Buffer

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'U'	'7'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'7'	STRING	-	-	1	1	ID Control : U_Clear_Buffer	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'7'						ID Control : U_Clear_Buffer		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	-	
							1	Clear Buffer Success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|6|T|U|7|*6126
ACK Respond	$ESP_OK|34|T|U|7|1|UART Clear Buffer Success|*6931
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - Control WIFI and Ethernet module connect to router

‘T’	‘I’	PL2	**Control WIFI and Ethernet module connect to router**	6
		‘0’	I_Start_WIFI_and_Connect_to_Router	
		‘1’	I_WIFI_Disconnect_to_Router	
		‘2’	I_Start_Ethernet_ENC28j60_and_Connect_to_Router	
		‘3’	I_Ethernet_ENC28j60_Disconnect_to_Router	
		‘4’	I_Start_Ethernet_W5XX_and_Connect_to_Router	
		‘5’	I_Ethernet_W5XX_Disconnect_to_Router	

HOST send Control I_Start_WIFI_and_Connect_to_Router

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘I’	‘0’	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	‘0’	STRING	-	-	1	1	ID Control : I_Start_WIFI_and_Connect_to_Router	

ESPrtk Respond

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit

PL2	'0'					ID Control : I_Start_WIFI_and_Connect_to_Router		
	PL3	INTEGER	-1	2	-	-	Status Control	
							-1	NACK: Invalid input /wrong format / out of range …?
							0	Error: No data SSID-PASS in Profile
							1	WiFi Connecting... please wait in 5 second
							2	WiFi Connected to router

Example

Message :	
HOST Send	$ESP_OK|6|T|I|0|*7A26
ACK Respond	$ESP_OK|72|T|I|0|1|WiFi Connecting... please wait in 5 second . List:[TEWM_B31B6B]|*4C32 >> Error timeout - Wifi cannot connect to router after 10s >> Wifi Connected to TEWM_B31B6B
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Tip: To change SSID-PASS , see UART_Configure → INTERNET configure.

HOST send Control I_WIFI_Disconnect_to_Router

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'1'	'1'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'1'	STRING	-	-	1	1	ID Control : I_WIFI_Disconnect_to_Router	

ESPrtk Respond

Payload	Type value	Min	Max	Min	Max	Describe	Unit

Index			Value	Value	Length	Length		
PL2	'1'						ID Control : I_WIFI_Disconnect_to_Router	
	PL3	INTEGER	-1	1	-	-	Status Control	
							-1	NACK: Invalid input /wrong format / out of range …?
							0	Disconnecting WiFi...Please wait in 3 seconds
							1	WiFi Disconnected success

Example

Message :	
HOST Send	$ESP_OK|6|T|I|1|*7B26
ACK Respond	$ESP_OK|26|T|I|1|1|WiFi Disconnected|*6845
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control I_Start_Ethernet_ENC28j60_and_Connect_to_Router

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'I'	'2'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'2'	STRING	-	-	1	1	ID Control : I_Start_Ethernet_ENC28j60_and_Connect_to_Router	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'2'						ID Control : I_Start_Ethernet_ENC28j60_and_Connect_to_Router		
	PL3	INTEGER	-1	2	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : Not found ENC28J60 on SPI port	
							1	Ethernet Connecting... please wait in 10 seconds	
							2	Ethernet Connected to Router	

Example

Message :	
HOST Send	$ESP_OK|6|T|I|2|*7826
ACK Respond	$ESP_OK|46|T|I|2|0|Error- Not found Ethernet on SPI port|*4962 $ESP_OK|57|T|I|2|1|Ethernet Connecting... please wait in 10 seconds|*3C4C $ESP_OK|27|T|I|2|2|Ethernet Connected|*376D
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control I_Ethernet_ENC28j60_Disconnect_to_Router

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'I'	'3'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'3'	STRING	-	-	1	1	ID Control : I_Ethernet_ENC28j60_Disconnect_to_Router	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'3'						ID Control : I_Ethernet_ENC28j60_Disconnect_to_Router		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Disconnecting Ethernet ...Please wait in 10 seconds	
							1	Ethernet Disconnected to router	

Example

Message :	
HOST Send	$ESP_OK|6|T|I|3|*7926
ACK Respond	$ESP_OK|60|T|I|3|0|Disconnecting Ethernet ...Please wait in 10 seconds|*6606 …. $ESP_OK|30|T|I|3|1|Ethernet Disconnected|*4D4F
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control I_Start_Ethernet_W5XX_and_Connect_to_Router

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'I'	'4'	Infor 1	Infor 2	Infor 3	...

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit

PL2	'4'	STRING	-	-	1	1	ID Control : I_Start_Ethernet_W5XX_and_Connect_to_Router	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'4'						ID Control : I_Start_Ethernet_W5XX_and_Connect_to_Router		
	PL3	INTEGER	-1	2	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : Not found W5XX module on SPI port	
							1	Ethernet Connecting... please wait in 10 seconds	
							2	Ethernet Connected to Router	

Example

Message :	
HOST Send	$ESP_OK|6|T|I|4|*7E26
ACK Respond	$ESP_OK|46|T|I|4|0|Error- Not found Ethernet on SPI port|*4F62 $ESPLOG|58|2|INFO|SPI Scan Success : Connected to Ethernet over SPI.|*5C31 >> Ethernet Connected $ESP_OK|27|T|I|4|2|Ethernet Connected|*316B
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control I_Ethernet_W5XX_Disconnect_to_Router

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'I'	'5'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'5'	STRING	-	-	1	1	ID Control : I_Ethernet_W5XX_Disconnect_to_Router	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'5'						ID Control : I_Ethernet_W5XX_Disconnect_to_Router		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Disconnecting Ethernet ...Please wait in 10 seconds	
							1	Ethernet Disconnected to router	

Example

Message :	
HOST Send	$ESP_OK|6|T|I|5|*7F26
ACK Respond	$ESP_OK|60|T|I|3|0|Disconnecting Ethernet ...Please wait in 10 seconds|*6606 …. $ESP_OK|30|T|I|5|1|Ethernet Disconnected|*4B4F
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - Control TCP Client Task communicate with Server

‘T’	‘T’	PL2	Control TCP Client Task communicate with Server	9
		‘0’	T_Connect_to_Server	
		‘1’	T_Check_Connected_to_Server	
		‘2’	T_Send_String	
		‘3’	T_Send_Binary	
		‘4’	T_Available	
		‘5’	T_Read_String	
		‘6’	T_Read_Binary	
		‘7’	T_Clear_Buffer	
		‘8’	T_Diconnect_to_Server	

HOST send Control T_Connect_to_Server

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'T'	'0'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'0'						ID Control : T_Connect_to_Server	
	PL3	STRING	-	-	1	200	TCP IP address of Server	
	PL4	INTEGER	1	65354	-	-	TCP Port number of Server	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'0'						ID Control : T_Connect_to_Server		
	PL3	INTEGER	-1	2	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: No WiFi/Ethernet connection	
							1	Error: TCP cannot connect to this IP	
							2	OK : TCP Connected success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :

HOST Send	$ESP_OK|20|T|T|0|google.com|80|*1D52
ACK Respond	$ESP_OK|50|T|T|0|2|TCP Connected , IP=google.com , Port = 80|*307D
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control T_Check_Connected_to_Server

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'T'	'1'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'1'	STRING	-	-	1	1	ID Control : T_Check_Connected_to_Server	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'1'						ID Control : T_Check_Connected_to_Server		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	TCP Client-No connection	
							1	TCP Client- Connected	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :

HOST Send	$ESP_OK|6|T|T|1|*6626
ACK Respond	$ESP_OK|37|T|T|1|1|TCP Client Connected success|*074F
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control T_Send_String

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'T'	'2'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'2'						ID Control : T_Send_String	
	PL3	STRING	-	-	1	1500	Data send in string format	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'2'						ID Control : T_Send_String		
	PL3	INTEGER	-1	2	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : Not connect to Server .	
							1	Error: Busy TX is sending	
							2	OK: Send String success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|29|T|T|2|GET / HTTP/1.1\r\n\r\n|*4470
ACK Respond	$ESP_OK|39|T|T|2|2|TCP Client Send String success|*7F08
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control T_Send_Binary

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'T'	'3'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'3'						ID Control : T_Send_Binary	
	PL3	HEX	-	-	2	1400	Data send in string format	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'3'						ID Control : T_Send_Binary		
	PL3	INTEGER	-1	2	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : Not connect to Server .	
							1	Error: Busy TX is sending	
							2	OK: Send Binary success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|43|T|T|3|474554202f20485454502F312F310D0A0D0A|*775C
ACK Respond	$ESP_OK|39|T|T|3|2|TCP Client Send Binary success|*6418
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control T_Available

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'T'	'4'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'4'	STRING	-	-	1	1	ID Control : T_Available	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'4'						ID Control : T_Available		
	PL3	INTEGER	-1	1000	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							[0,1000]	Size available RX buffer TCP	

Example

Message :	
HOST Send	$ESP_OK|6|T|T|4|*6326
ACK Respond	$ESP_OK|11|T|T|4|5744|*2B5E
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control T_Read_String

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'T'	'5'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'5'						ID Control : T_Read_String	
	PL3	INTEGER	1	1000	-	-	Size read buffer TCP	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'5'						ID Control : T_Read_String		
	PL3	INTEGER	-1	1000	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							[0,1000]	Size of string respond	
	PL4	STRING	-	-	0	1000	Data read buffer in String format		

Example

Message :	
HOST Send	$ESP_OK|9|T|T|5|60|*1701
ACK Respond	$ESP_OK|70|T|T|5|60|abs: -1 Cache-Control: private, max-age=0 Content-Type: text|*123A
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control T_Read_Binary

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'T'	'6'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'6'						ID Control : T_Read_Binary	
	PL3	INTEGER	1	1000	-	-	Size read buffer TCP	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'6'						ID Control : T_Read_Binary		
	PL3	INTEGER	-1	1000	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							[0,1000]	Size of string respond	
	PL4	STRING	-	-	-	-	Data read buffer in HEX format		

Example

Message :	
HOST Send	$ESP_OK|9|T|T|6|60|*1402
ACK Respond	$ESP_OK|130|T|T|6|60| 485454502F312E3120323030204F4B0D0A446174653A204D6F6E2C203230204A616E20323032 302031333A30373A313620474D540D0A457870697265|*685F
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control T_Clear_Buffer

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'T'	'7'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'7'	STRING	-	-	1	1	ID Control : T_Clear_Buffer	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'7'						ID Control : T_Clear_Buffer		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	-	
							1	Clear Buffer Success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|6|T|T|7|*6026
ACK Respond	$ESP_OK|40|T|T|7|1|TCP Client Clear Buffer Success|*273D
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control T_Diconnect_to_Server

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'T'	'8'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'8'	STRING	-	-	1	1	ID Control : T_Diconnect_to_Server	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'8'						ID Control : T_Diconnect_to_Server		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	-	
							1	TCP Disconnected Success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|6|T|T|8|*6F26
ACK Respond	$ESP_OK|25|T|T|8|1|TCP Disconnected|*2952
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Write Configure - Control Bluetooth Classic and Bluetooth BLE on ESPrtk

‘T’	‘B’	PL2	**Control Bluetooth Classic and Bluetooth BLE on ESPrtk**	8
		‘0’	B_Start_Bluetooth	
		‘1’	B_Check_Client_Connected	
		‘2’	B_Send_String	
		‘3’	B_Send_Binary	
		‘4’	B_Available	
		‘5’	B_Read_String	
		‘6’	B_Read_Binary	
		‘7’	B_Clear_Buffer	

HOST send Control B_Start_Bluetooth

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘B’	‘0’	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	‘0’						ID Control : B_Start_Bluetooth	

	PL3	INTEGER	0	1	-	-	Type Bluetooth		
							0	Start Bluetooth Classic	
							1	Start Bluetooth Low Energy (BLE)	
	PL4	STRING	-	-	1	100	Name of ESPrtk Bluetooth		

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'0'						ID Control : B_Start_Bluetooth		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: Bluetooth start Failed	
							1	OK : Bluetooth Started success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :		
HOST Send	Start Bluetooth Classic	$ESP_OK|22|T|B|0|0|my_BLT_device|*216B
	Start Bluetooth BLE	$ESP_OK|22|T|B|0|1|my_BLE_device|*317A
ACK Respond	$ESP_OK|62|T|B|0|1|Bluetooth Classic Started success,Name: my_BLT_device|*2525 $ESP_OK|58|T|B|0|1|Bluetooth BLE Started success,Name: my_BLE_device|*320D	
NACK Respond		
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…	

HOST send Control B_Check_Client_Connected

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'B'	'1'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'1'	STRING	-	-	1	1	ID Control : B_Check_Client_Connected	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'1'						ID Control : B_Check_Client_Connected		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	No Bluetooth client connect to ESPrtk	
							1	Has Bluetooth client connected to ESPrtk	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|6|T|B|1|*7026
ACK Respond	$ESP_OK|32|T|B|1|0|Bluetooth-No connection|*1B57 $ESP_OK|36|T|B|1|1|Bluetooth Connected success|*1804
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control B_Send_String

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'B'	'2'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'2'						ID Control : B_Send_String	
	PL3	STRING	-	-	1	1500	Data send in string format	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'2'						ID Control : B_Send_String		
	PL3	INTEGER	-1	2	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : No Bluetooth client connect to ESPrtk or Bluetooth was not started yet.	
							1	Error: Busy TX is sending	
							2	OK: Send String success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|38|T|B|2|Hello Bluetooth.\r\n I'm ESPrtk|*1C44
ACK Respond	$ESP_OK|38|T|B|2|2|Bluetooth Send String success|*604D
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control B_Send_Binary

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	PL5	…

$ESP_OK|xx|	'T'	'B'	'3'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'3'						ID Control : B_Send_Binary	
	PL3	HEX	-	-	2	1400	Data send in string format	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'3'						ID Control : B_Send_Binary		
	PL3	INTEGER	-1	2	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : No Bluetooth client connect to ESPrtk or Bluetooth was not started yet.	
							1	Error: Busy TX is sending	
							2	OK: Send Binary success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|67|T|B|3|48656c6c6f20426c7565746f6f74682e0D0a49276d2045535072746b0D0a|*6749
ACK Respond	$ESP_OK|38|T|B|3|2|Bluetooth Send Binary success|*7B5C
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control B_Available

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'B'	'4'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'4'	STRING	-	-	1	1	ID Control : B_Available	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'4'						ID Control : B_Available		
	PL3	INTEGER	-1	1000	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							[0,1000]	Size available RX buffer Bluetooth	

Example

Message :	
HOST Send	$ESP_OK|6|T|B|4|*7526
ACK Respond	$ESP_OK|9|T|B|4|31|*0413
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control B_Read_String

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'B'	'5'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'5'						ID Control : B_Read_String	
	PL3	INTEGER	1	1000	-	-	Size read buffer Bluetooth	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'5'						ID Control : B_Read_String		
	PL3	INTEGER	-1	1000	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							[0,1000]	Size of string respond	
	PL4	STRING	-	-	0	1000	Data read buffer in String format		

Example

Message :	
HOST Send	$ESP_OK|10|T|B|5|100|*0E3A
ACK Respond	$ESP_OK|40|T|B|5|31|Hi ESPrtk, Im mobile device. |*705B
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control B_Read_Binary

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'B'	'6'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'6'						ID Control : B_Read_Binary	
	PL3	INTEGER	1	1000	-	-	Size read buffer RX Bluetooth	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'6'						ID Control : B_Read_Binary		
	PL3	INTEGER	-1	1000	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							[0,1000]	Size of string respond	
	PL4	STRING	-	-	-	-	Data read buffer in HEX format		

Example

Message :	
HOST Send	$ESP_OK|10|T|B|6|100|*0D3A
ACK Respond	$ESP_OK|72|T|B|6|31| 48692045535072746B2C20496D206D6F62696C6520646576696365 2E200D0A|*3D0C
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control B_Clear_Buffer

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'B'	'7'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'7'	STRING	-	-	1	1	ID Control : B_Clear_Buffer	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'7'						ID Control : B_Clear_Buffer		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	-	
							1	Clear Buffer Success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|6|T|B|7|*7626
ACK Respond	$ESP_OK|39|T|B|7|1|Bluetooth Clear Buffer Success|*3767
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Hardware Control – Control OLED Display (SSD1306/SH1106)

HOST send Control OLED Display (SSD1306/SH1106)

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘O’	ID Control	Infor 1	Infor 2	Infor 3	...

‘T’	‘O’	PL2	Control OLED Display (SSD1306/SH1106)	12
		‘0’	O_Start_OLED	
		‘1’	O_Update_and_Display	
		‘2’	O_Clear_Display	
		‘3’	O_Draw_Dot	
		‘4’	O_Draw_Line	
		‘5’	O_Draw_Rect	
		‘6’	O_Draw_Rect_Corner	
		‘7’	O_Draw_Triangle	
		‘8’	O_Draw_Circle	
		‘9’	O_Draw_Elippse	
		‘A’	O_Draw_Character	
		‘B’	O_Draw_Number	

HOST send Control O_Start_OLED

<Header> +	<Payload>

<Payload_length>	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘O’	‘0’	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘0’						ID Control : O_Start_OLED		
	PL3	INTEGER	0	1	-	-	Type OLED Display		
							0	OLED SH1106 128x64	
							1	OLED SSD1306 128x64	
	PL4	INTEGER	1	255	-	-	Brighnesss		
	PL5	INTEGER	1	255	-	-	Contrast		
	PL6	INTEGER	0	1	-	-	Vertical Flip Screen		
							0	No	
							1	Yes	
	PL7	INTEGER	0	1	-	-	Flag Auto Update Display without using $ESP_OK|6|T|O|1|*7D26 command (O_Update_and_Display)		
							0	No	
							1	Yes	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘0’						ID Control : O_Start_OLED		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : not found OLED Display on I2C port	
							1	OK : Started OLED success	

	PL4	STRING	-	-	-	-	Describe status respond control as text	

Example

Message :	
HOST Send	$ESP_OK|20|T|O|0|1|250|250|0|0|*0509
ACK Respond	$ESP_OK|101|T|O|0|1|Start OLED Success, Type: SSD1306,Brightness = 250,Contrast = 250, Flip = 0 ,Auto Update = 0|*223C
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Update_and_Display

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'O'	'1'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'1'	STRING	-	-	1	1	ID Control : O_Update_and_Display	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'1'						ID Control : O_Update_and_Display		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: OLED Display was not started yet.	
							1	OK: (OLED-Updated Success) Done	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|6|T|O|1|*7D26
ACK Respond	$ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Clear_Display

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'O'	'2'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'2'						ID Control : O_Clear_Display		
	PL3	INTEGER	0	127	-	-	X Clear		
	PL4	INTEGER	0	63	-	-	Y Clear		
	PL5	INTEGER	0	128	-	-	Width Clear		pixel
	PL6	INTEGER	0	64	-	-	Height Clear		pixel
	PL7	INTEGER	0	1	-	-	Color clear		
							0	BLACK	
							1	WHITE	

ESPrtk Respond

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit

PL2	'2'					ID Control : O_Clear_Display		
	PL3	INTEGER	-1	1	-	-	Status Control	
							-1	NACK: Invalid input /wrong format / out of range …?
							0	Error: OLED Display was not started yet.
							1	OK: OLED-Clear) Done
	PL4	STRING	-	-	-	-	Describe status respond control as text	

Example

Message :	
HOST Send	$ESP_OK|19|T|O|2|0|0|128|64|0|*3509 $ESP_OK|6|T|O|1|*7D26 (need add this command to update display if Flag Auto Update Display is turn off)
ACK Respond	$ESP_OK|26|T|O|2|1|(OLED-Clear) Done|*0A5C $ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Draw_Dot

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'O'	'3'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'3'						ID Control : O_Draw_Dot		
	PL3	INTEGER	-32768	32767	-	-	X		
	PL4	INTEGER	-32768	32767	-	-	Y		
	PL5	INTEGER	0	1	-	-	Color Display		
							0	BLACK	

							1	WHITE	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'3'						ID Control : O_Draw_Dot		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: OLED Display was not started yet.	
							1	OK: Draw Dot done	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|14|T|O|3|64|32|1|*0271 $ESP_OK|6|T|O|1|*7D26 (need add this command to update display if Flag Auto Update Display is turn off)
ACK Respond	$ESP_OK|29|T|O|3|1|(OLED-Draw Dot) Done|*0233 $ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Draw_Line

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'O'	'4'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'4'						ID Control : O_Draw_Line		
	PL3	INTEGER	-32768	32767	-	-	XA		
	PL4	INTEGER	-32768	32767	-	-	YA		
	PL5	INTEGER	-32768	32767	-	-	XB		
	PL6	INTEGER	-32768	32767	-	-	YB		
	PL7	INTEGER	0	1	-	-	Color Display		
							0	BLACK	
							1	WHITE	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'4'						ID Control : O_Draw_Line		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: OLED Display was not started yet.	
							1	OK: Draw Line Done	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|18|T|O|4|5|5|32|20|1|*0976 $ESP_OK|6|T|O|1|*7D26 (need add this command to update display if Flag Auto Update Display is turn off)
ACK Respond	$ESP_OK|30|T|O|4|1|(OLED-Draw Line) Done|*7C77 $ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	

Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Draw_Rect

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘O’	‘5’	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘5’						ID Control : O_Draw_Rect		
	PL3	INTEGER	-32768	32767	-	-	X		
	PL4	INTEGER	-32768	32767	-	-	Y		
	PL5	INTEGER	-32768	32767	-	-	Width Rect		pixel
	PL6	INTEGER	-32768	32767	-	-	Height Rect		pixel
	PL7	INTEGER	0	1	-	-	Color Display		
							0	BLACK	
							1	WHITE	
	PL8	INTEGER	0	1	-	-	Fill object draw		
							0	No	
							1	Yes	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘5’						ID Control : O_Draw_Rect		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: OLED Display was not started yet.	
							1	OK: Draw Rectangle Done	

	PL4	STRING	-	-	-	-	Describe status respond control as text	

Example

Message :	
HOST Send	$ESP_OK|22|T|O|5|50|20|40|20|1|0|*4F75 $ESP_OK|6|T|O|1|*7D26 (need add this command to update display if Flag Auto Update Display is turn off)
ACK Respond	$ESP_OK|35|T|O|5|1|(OLED-Draw Rectangle) Done|*1745 $ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Draw_Rect_Corner

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'O'	'6'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'6'						ID Control : O_Draw_Rect_Corner		
	PL3	INTEGER	-32768	32767	-	-	X		
	PL4	INTEGER	-32768	32767	-	-	Y		
	PL5	INTEGER	-32768	32767	-	-	Width Rectangle Corner		pixel
	PL6	INTEGER	-32768	32767	-	-	Height Rectangle Corner		pixel
	PL7	INTEGER	-32768	32767	-	-	Radius Corner		
	PL8	INTEGER	0	1	-	-	Color Display		
							0	BLACK	
							1	WHITE	
	PL9	INTEGER	0	1	-	-	Fill object draw		

							0	No	
							1	Yes	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'6'						ID Control : O_Draw_Rect_Corner		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: OLED Display was not started yet.	
							1	OK: Draw Rectangle Corner Done	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|24|T|O|6|12|20|40|20|5|1|0|*050B $ESP_OK|6|T|O|1|*7D26 (need add this command to update display if Flag Auto Update Display is turn off)
ACK Respond	$ESP_OK|42|T|O|6|1|(OLED-Draw Rectangle Corner) Done|*1354 $ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Draw_Triangle

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'O'	'7'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'7'						ID Control : O_Draw_Triangle		
	PL3	INTEGER	-32768	32767	-	-	XA		
	PL4	INTEGER	-32768	32767	-	-	YA		
	PL5	INTEGER	-32768	32767	-	-	XB		
	PL6	INTEGER	-32768	32767	-	-	YB		
	PL7	INTEGER	-32768	32767	-	-	XC		
	PL8	INTEGER	-32768	32767	-	-	YC		
	PL9	INTEGER	0	1	-	-	Color Display		
							0	BLACK	
							1	WHITE	
	PL10	INTEGER	0	1	-	-	Fill object draw		
							0	No	
							1	Yes	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'7'						ID Control : O_Draw_Triangle		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: OLED Display was not started yet.	
							1	OK: Draw Triangle Done	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|28|T|O|7|10|10|40|10|25|30|1|0|*470F

	$ESP_OK|6|T|O|1|*7D26 (need add this command to update display if Flag Auto Update Display is turn off)
ACK Respond	$ESP_OK|34|T|O|7|1|(OLED-Draw Triangle) Done|*7B6A $ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Draw_Circle

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'O'	'8'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'8'						ID Control : O_Draw_Circle		
	PL3	INTEGER	-32768	32767	-	-	X Center circle		
	PL4	INTEGER	-32768	32767	-	-	Y Center circle		
	PL5	INTEGER	-32768	32767	-	-	Circle radius		pixel
	PL6	INTEGER	0	1	-	-	Color Display		
							0	BLACK	
							1	WHITE	
	PL	INTEGER	0	1	-	-	Fill object draw		
							0	No	
							1	Yes	

ESPrtk Respond

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit

PL2	'8'					ID Control : O_Draw_Circle		
	PL3	INTEGER	-1	1	-	-	Status Control	
							-1	NACK: Invalid input /wrong format / out of range …?
							0	Error: OLED Display was not started yet.
							1	OK: Draw Circle Done
	PL4	STRING	-	-	-	-	Describe status respond control as text	

Example

Message :	
HOST Send	$ESP_OK|19|T|O|8|64|32|20|1|0|*360A $ESP_OK|6|T|O|1|*7D26 (need add this command to update display if Flag Auto Update Display is turn off)
ACK Respond	$ESP_OK|32|T|O|8|1|(OLED-Draw Circle) Done|*6E08 $ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Draw_Elippse

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'O'	'9'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'9'						ID Control : O_Draw_Elippse	
	PL3	INTEGER	-32768	32767	-	-	X Elippse's Center	
	PL4	INTEGER	-32768	32767	-	-	Y Elippse's Center	
	PL5	INTEGER	-32768	32767	-	-	Width Radius	pixel
	PL6	INTEGER	-32768	32767	-	-	Height Radius	pixel

	PL7	INTEGER	0	1	-	-	Color Display		
							0	BLACK	
							1	WHITE	
	PL8	INTEGER	0	1	-	-	Fill object draw		
							0	No	
							1	Yes	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'9'						ID Control : O_Draw_Elippse		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: OLED Display was not started yet.	
							1	OK: Draw Done	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|22|T|O|9|64|32|40|20|1|0|*4770 $ESP_OK|6|T|O|1|*7D26 (need add this command to update display if Flag Auto Update Display is turn off)
ACK Respond	$ESP_OK|33|T|O|9|1|(OLED-Draw Elippse) Done|*0A33 $ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Draw_Character

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'O'	'A'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'A'						ID Control : O_Draw_Character		
	PL3	INTEGER	-32768	32767	-	-	X String		
	PL4	INTEGER	-32768	32767	-	-	Y String		
	PL5	INTEGER	0	1	-	-	Color Display		
							0	BLACK	
							1	WHITE	
	PL6	STRING	-	-	1	1000	Data draw as text string format		

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'A'						ID Control : O_Draw_Character		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: OLED Display was not started yet.	
							1	OK: Draw Char Done	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :

HOST Send	$ESP_OK|59|T|O|A|80|2|1|Hello\nESPrtk\nOLED\nTest\nDraw\nString\nDone|*5478 $ESP_OK|6|T|O|1|*7D26 (need add this command to update display if Flag Auto Update Display is turn off)
ACK Respond	$ESP_OK|30|T|O|A|1|(OLED-Draw Char) Done|*1F77 $ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control O_Draw_Number

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'O'	'B'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'B'						ID Control : O_Draw_Number	
	PL3	INTEGER	-32768	32767	-	-	X	
	PL4	INTEGER	-32768	32767	-	-	Y	
	PL5	INTEGER	0	2	-	-	Font text draw number string 0 ASCII_5x7 1 CASIO_12x16 2 STYLE_16x16	
	PL6	INTEGER	0	1	-	-	Color Display 0 BLACK 1 WHITE	
	PL7	STRING	-	-	-	-	Data draw as text number format	

ESPrtk Respond

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit

PL2	'B'					ID Control : O_Draw_Number	
	PL3	INTEGER	-1	1	-	-	Status Control
						-1	NACK: Invalid input /wrong format / out of range …?
						0	Error: OLED Display was not started yet.
						1	OK: Draw Done
	PL4	STRING	-	-	-	-	Describe status respond control as text

Example

Message :	
HOST Send	$ESP_OK|18|T|O|B|5|2|0|1|123|*7B38 $ESP_OK|20|T|O|B|5|12|1|1|-2.3|*7244 $ESP_OK|19|T|O|B|5|32|2|1|0.3|*566E $ESP_OK|6|T|O|1|*7D26 (need add this command to update display if Flag Auto Update Display is turn off)
ACK Respond	$ESP_OK|32|T|O|B|1|(OLED-Draw Number) Done|*0513 $ESP_OK|27|T|O|1|1|(OLED-Update) Done|*6010
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Hardware Control – SD Card

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘S’	ID Control	Infor 1	Infor 2	Infor 3	...

‘T’	‘S’	PL2	Control SD Card	10
		‘0’	S_Start_SD_Card	
		‘1’	S_Check_Folder_Exist	
		‘2’	S_Create_Folder	
		‘3’	S_Delete_File	
		‘4’	S_Rename_File	
		‘5’	S_Check_File_Exist	
		‘6’	S_Create_File	
		‘7’	S_Get_Size_File	
		‘8’	S_Read_Binary_data_File	
		‘9’	S_Append_Binary_data_to_File	

HOST send Control S_Start_SD_Card

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘S’	‘0’	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'0'	STRING	-	-	1	1	ID Control : S_Start_SD_Card	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'0'						ID Control : S_Start_SD_Card		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : Card Mount Failed / Not found SD Card on SPI port	
							1	OK : Start SD Card success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|6|T|S|0|*6026
ACK Respond	$ESP_OK|67|T|S|0|1|Start SD Card Success , Card type:SD , Card size: 1802 MB |*485A
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control S_Check_Folder_Exist

<Header> +	<Payload>

<Payload_length>	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘S’	‘1’	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	‘1’						ID Control : S_Check_Folder_Exist	
	PL3	STRING	-	-	1	1500	Path Folder	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘1’						ID Control : S_Check_Folder_Exist		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	OK: Not exist (Not found this folder)	
							1	OK: Existed	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|15|T|S|1|/my_test|*5D08
ACK Respond	$ESP_OK|16|T|S|1|1|Existed|*3318
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control S_Create_Folder

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'S'	'2'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'2'						ID Control : S_Create_Folder	
	PL3	STRING	-	-	3	1500	Path of folder	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'2'						ID Control : S_Create_Folder		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : Create Folder failed	
							1	OK: Create Folder Success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|15|T|S|2|/my_test|*5E0B
ACK Respond	$ESP_OK|30|T|S|2|1|Create Folder Success|*3708
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control S_Delete_File

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'S'	'3'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'3'						ID Control : S_Delete_File	
	PL3	STRING	-	-	3	1500	Path of file	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'3'						ID Control : S_Delete_File		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : Error: Delete File failed	
							1	OK : Delete File Success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|21|T|S|3|/my_test/i.txt|*4877
ACK Respond	$ESP_OK|28|T|S|3|1|Delete File Success|*327F
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control S_Rename_File

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'S'	'4'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'4'						ID Control : S_Rename_File	
	PL3	STRING	-	-	2	1500	Path file A	
	PL4	STRING	-	-	2	1500	Path file B	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'4'						ID Control : S_Rename_File		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : Rename failed	
							1	OK : File renamed success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :

HOST Send	Rename file i.txt to j.txt $ESP_OK|36|T|S|4|/my_test/i.txt|/my_test/j.txt|*546A
ACK Respond	$ESP_OK|29|T|S|4|1|File renamed success|*594F
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control S_Check_File_Exist

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'S'	'5'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'5'						ID Control : S_Check_File_Exist	
	PL3	STRING	-	-	1	1500	Path of file	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'5'						ID Control : S_Check_File_Exist		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	Error : File path invalid or file not exist	
							0	Error : Not exist - File not found	
							1	OK: File Existed	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|21|T|S|5|/my_test/i.txt|*4E71
ACK Respond	$ESP_OK|16|T|S|5|1|Existed|*3718
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control S_Create_File

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'S'	'6'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'6'						ID Control : S_Create_File	
	PL3	STRING	-	-	1	1500	Path of file create	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'6'						ID Control : S_Create_File		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: Create File failed	
							1	OK: Create File Success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|21|T|S|6|/my_test/i.txt|*4D72
ACK Respond	$ESP_OK|28|T|S|6|1|Create File Success|*2A6C
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control S_Get_Size_File

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'S'	'7'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'7'						ID Control : S_Get_Size_File	
	PL3	STRING	-	-	1	1500	Path of	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value			Describe		Unit
PL2	'7'						ID Control : S_Get_Size_File		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error : Failed open file for read size	
							1	OK	
	PL4	INTEGER	0	10000000000	-	-	Size of file		byte

Example

Message :	
HOST Send	$ESP_OK|21|T|S|7|/my_test/i.txt|*4C73
ACK Respond	$ESP_OK|12|T|S|7|1|241|*5442
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control S_Read_Binary_data_File

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘S’	‘8’	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	‘8’						ID Control : S_Read_Binary_data_File	
	PL3	STRING	-	-	1	1500	Path of file	
	PL4	INTEGER	0	10000000000	-	-	Offset index start read file	
	PL5	INTEGER	1	1000	-	-	Size read	byte

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value			Describe		Unit
PL2	‘8’						ID Control : S_Read_Binary_data_File		
	PL3	INTEGER	-1	1000	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	

							[0,1000]	Size of data read respond	
	PL4	STRING	-	-	-	-	Data read in HEX format		

Example

Message :	
HOST Send	$ESP_OK|27|T|S|8|/my_test/i.txt|0|500|*404A
ACK Respond	$ESP_OK|493|T|S|8|241| 45535072746B2056657273696F6E3A20322E372E39202E0D0A45535072746B2049443A2038313930395F4748423941202E0D0A5265636F726420696E20616374696F6E2061733A2055415254204036F6E666967757265202D3E204861726457761726520436F6E74726F6C202D3E2053442043617264202E0D0A436F6E7461637420537570706F72743A20687474703A2F2F65737072746B2E776F726470726573732E636F6D202C20456D61696C203A65737072746B40676D61696C2E636F6D202C20544D3A207068756E6774686169736F6E393640676D61696C2E636F6D202E0D0A48656C6C6F2045535072746B0D0A|*5835
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Control S_Append_Binary_data_to_File

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'S'	'9'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'9'						ID Control : S_Append_Binary_data_to_File	
	PL3	STRING	-	-	1	1500	Path of file	
	PL4	HEX	-	-	1	1400	Append data to file in HEX format	

ESPrtk Respond

Payload	Type value	Min	Max	Min	Max	Describe	Unit

Index			Value	Value	Length	Length		
PL2	'9'						ID Control : S_Append_Binary_data_to_File	
	PL3	INTEGER	-1	1	-	-	Status Control	
							-1	NACK: Invalid input /wrong format / out of range …?
							0	Error : Failed open file for append data
							1	OK : Append Data success
	PL4	STRING	-	-	-	-	Describe status respond control as text	

Example

Message :	
HOST Send	$ESP_OK|50|T|S|9|/my_test/i.txt|48656C6C6F2045535072746B0D0A|*3040
ACK Respond	$ESP_OK|33|T|S|9|1|File Append Data success|*1568
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Hardware Control – IMU (MPU9250)

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'M'	ID Control	Infor 1	Infor 2	Infor 3	...

'T'	'M'	PL2	Control IMU (MPU9250)	5
		'0'	M_Start_Print_IMU_Data	
		'1'	M_Stop_Print_IMU_Data	
		'2'	M_Start_IMU_Calibrate	
		'3'	M_Stop_IMU_Calibrate	
		'4'	M_Save_Calibrate_Result	
			Disable/Enable type print IMU	

HOST send M_Start_Print_IMU_Data

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'M'	'0'	Infor 1	Infor 2	Infor 3	...

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit

PL2	‘0’						ID Control :M_Start_Print_IMU_Data	
	PL3	INTEGER	0	9	-	-	Type frequency update IMU	
							0	2 Hz
							1	10 Hz
							2	15 Hz
							3	20 Hz
							4	25 Hz
							5	35 Hz
							6	50 Hz
							7	100 Hz
							8	150 Hz
							9	200 Hz

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘0’						ID Control : M_Start_Print_IMU_Data		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error: Not found data configure for MPU9250 in Profile	
							1	OK: Is starting connect to IMU and print IMU data	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|8|T|M|0|1|*3D5A
ACK Respond	……. $ESP_OK|64|T|M|0|1|Is starting print IMU data , Type Frequency update = 1 |*1C0E ……..

	$ESPIM1|81|82.32|199.63|172.13|-2.40|-7.20|-50.41|0.54|-0.52|-0.57|29.17|-2.40|-7.20|-50.41|*186D $ESPIM2|82|0.35|-0.25|-0.88|0.20|136.90|-141.23|-31.31|21.44|||||139.56|-141.49|-30.51|22.45|*750C $ESPIM1|84|178.90|116.04|202.62|17.55|-21.91|-30.16|0.57|-0.76|-0.12|29.16|17.55|-21.91|-30.16|*2071 $ESPIM2|82|0.57|-0.51|-0.57|0.30|79.64|-100.98|-20.24|334.15|||||83.66|-106.79|-20.59|315.65|*7D05 $ESPIM1|82|117.38|127.38|185.91|37.51|-10.50|-8.85|-0.09|-0.79|0.14|29.16|37.51|-10.50|-8.85|*2211 $ESPIM2|77|0.66|-0.60|-0.20|0.40|51.81|-78.48|12.85|316.72|||||52.77|-79.97|6.36|316.71|*4774 ……..
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send M_Stop_Print_IMU_Data

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘M’	‘1’	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	‘1’	STRING	-	-	1	1	ID Control : M_Stop_Print_IMU_Data	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘1’						ID Control : M_Stop_Print_IMU_Data		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error:	
							1	OK: Stopped Print IMU data	

	PL4	STRING	-	-	-	-	Describe status respond control as text	

Example

Message :	
HOST Send	$ESP_OK|6|T|M|1|*7F26
ACK Respond	$ESP_OK|31|T|M|1|1|Stopped Print IMU data|*031B
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send Disable/Enable type print IMU

Please see “Control Enable Type Print on TX_UART0” → PL3 → BIT11 and BIT12.
Example:

```
$ESP_OK|19|T|P|0|000000000000|*2157  -> Disable Print ALL output UART_0
$ESP_OK|19|T|P|0|111111111100|*2156  -> Disable Print IMU 1 + IMU2
$ESP_OK|19|T|P|0|111111111110|*2057  ->  Disable Print  IMU 1 , Enable Print IMU2
$ESP_OK|19|T|P|0|111111111101|*2056  ->  Enable Print IMU1  , Disable Print  IMU 2
$ESP_OK|19|T|P|0|111111111111|*2157  ->  Enable Print ALL output UART_0
```

HOST send M_Start_IMU_Calibrate

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘M’	‘2’	Infor 1	Infor 2	Infor 3	...

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit

PL2	'2'	STRING	-	-	1	1	ID Control : M_Start_IMU_Calibrate	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'2'						ID Control : M_Start_IMU_Calibrate		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	**Error:** Not found data configure for MPU9250 in Profile	
							1	OK: Is starting connect to IMU and print IMU data	
							2	**Error** : Low memory - Fail to malloc new buffer !	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|6|T|M|2|*7C26
ACK Respond	$ESP_OK|30|T|M|2|1|Started IMU Calibrate|*2F54
Data Respond	$ESPLOG|59|2|INFO|CAL|Please rotate the sensor to the different angle|*6224 $ESPLOG|59|2|INFO|CAL|Please rotate the sensor to the different angle|*6224 $ESPLOG|74|2|INFO|CAL|Surface scan (0.00 %),Input [-4.2010503,4.0510130,-69.0172577]|*6C01 $ESPLOG|74|2|INFO|CAL|Surface scan (0.01 %),Input [-4.2010503,4.0510130,-69.0172577]|*6D01 $ESPLOG|74|2|INFO|CAL|Surface scan (0.02 %),Input [-9.7524385,5.1012754,-67.9669952]|*6E07 $ESPLOG|75|2|INFO|CAL|Surface scan (0.03 %),Input [-17.5543880,5.8514628,-67.9669952]|*5C76 $ESPLOG|75|2|INFO|CAL|Surface scan (0.04 %),Input [-24.4561138,5.8514628,-62.7156792]|*5276 $ESPLOG|75|2|INFO|CAL|Surface scan (0.05 %),Input [-31.9579887,6.7516880,-58.0645180]|*527A …… ……
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send M_Stop_IMU_Calibrate

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'M'	'3'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'3'	STRING	-	-	1	1	ID Control : M_Stop_IMU_Calibrate	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'3'						ID Control : M_Stop_IMU_Calibrate		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Is calculating ... , please wait.	
							1	**OK**: Stopped IMU Calibrate. (Success)	
							2	**Error** : Incomplete samples (We need more samples to be able to calibrate!	
							3	**Error** : Incomplete samples (We need more samples on the X and Y axes by rotating the sensor around the Z axis!)	
							4	**Error** : Fail to calculating , please try again !	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :

HOST Send	$ESP_OK|6|T|M|3|*7D26
ACK Respond	$ESP_OK|41|T|M|3|0|Is calculating ... , please wait|*0214 …. (Calibrate progress)… …. (Noise filter progress)… …. (Printout data calibrated progress)… …. (Printout Bias+Rotations matrices)… $ESP_OK|30|T|M|3|1|Stopped IMU Calibrate|*2656
Data Respond	$ESP_OK|41|T|M|3|0|Is calculating ... , please wait|*0214 …….. …….. $ESPLOG|27|2|INFO|CAL|Calibrating:1 %|*396B $ESPLOG|27|2|INFO|CAL|Calibrating:2 %|*3A6B $ESPLOG|27|2|INFO|CAL|Calibrating:3 %|*3B6B …….. $ESPLOG|28|2|INFO|CAL|Calibrating:90 %|*0E23 $ESPLOG|28|2|INFO|CAL|Calibrating:95 %|*0B23 $ESPLOG|29|2|INFO|CAL|Calibrating:100 %|*3755 …….. …….. …….. $ESPLOG|28|2|INFO|CAL|Noise filter:0 %|*0B2F $ESPLOG|28|2|INFO|CAL|Noise filter:3 %|*082F $ESPLOG|28|2|INFO|CAL|Noise filter:6 %|*0D2F …….. $ESPLOG|29|2|INFO|CAL|Noise filter:90 %|*3355 $ESPLOG|29|2|INFO|CAL|Noise filter:93 %|*3055 $ESPLOG|29|2|INFO|CAL|Noise filter:96 %|*3555 …….. …….. …….. $ESPLOG|69|2|INFO|CAL|Count (1/617),Output [-24.6396484,26.2568359,-19.2402344]|*112C $ESPLOG|69|2|INFO|CAL|Count (2/617),Output [-24.6396484,26.2568359,-19.2402344]|*122F $ESPLOG|69|2|INFO|CAL|Count (3/617),Output [-22.3652344,25.3867188,-21.1152344]|*102C …….. $ESPLOG|70|2|INFO|CAL|Count (615/617),Output [12.0615234,22.5830078,-25.9902344]|*3D36 $ESPLOG|70|2|INFO|CAL|Count (616/617),Output [14.1621094,25.6347656,-25.7070312]|*323C $ESPLOG|70|2|INFO|CAL|Count (617/617),Output [12.6474609,24.3251953,-24.7656250]|*3E3D …….. …….. …….. $ESPLOG|205|2|INFO|CAL|Success ! Bias 1x3 [-11.4422918 , -23.6864032 , -40.1461467] ,Rotation 3x3 [1.0044592 , -0.0014763 , -0.0122153 , -0.0014784 , 0.9697963 , -0.0028070 , -0.0122177 , -

	0.0028135 , 1.0257844] |*1007 …….. …….. …….. $ESP_OK|30|T|M|3|1|Stopped IMU Calibrate|*2656
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send M_Save_Calibrate_Result

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	'T'	'M'	'4'	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit
PL2	'4'	STRING	-	-	1	1	ID Control : M_Save_Calibrate_Result	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'4'						ID Control : M_Save_Calibrate_Result		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	**Error:**	
							1	OK: Updated Calibrate Result	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|6|T|M|4|*7A26
ACK Respond	$ESP_OK|33|T|M|4|1|Updated Calibrate Result|*4650
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

Command Hardware Control – Deep Sleep

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘V’	ID Control	Infor 1	Infor 2	Infor 3	...

‘T’	‘V’	PL2	Control Deep Sleep	2
		‘0’	V_Enter_Deep_Sleep_mode	
		‘1’	V_Exit_Deep_Sleep_mode_and_WakeUp	

HOST send V_Enter_Deep_Sleep_mode

<Header> + <Payload_length>	<Payload>						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘V’	‘0’	Infor 1	Infor 2	Infor 3	...

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘1’	STRING	-	-	1	1	ID Control : V_Enter_Deep_Sleep_mode		
PL3		STRING	0	1	1	1	0	Only ESPrtk (ESP32) enter Deep Sleep mode .	
							1	ESPrtk (ESP32) and all other modules connect to ESPrtk will enter Deep Sleep mode .	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	‘1’						ID Control : V_Enter_Deep_Sleep_mode		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	Error:	
							1	OK: Enter Deep Sleep mode success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
HOST Send	$ESP_OK|8|T|V|0|1|*265A
ACK Respond	$ESP_OK|38|T|V|0|1|Enter Deep Sleep mode success|*3A40
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

HOST send V_Exit_Deep_Sleep_mode_and_WakeUp

<Header> + <Payload_length>	**<Payload>**						
	PL0	PL1	PL2	PL3	PL4	PL5	…
$ESP_OK|xx|	‘T’	‘V’	‘1’	Infor 1	Infor 2	Infor 3	...

Payload Index	Type value	Min Value	Max Value	Min Length	Max Length	Describe	Unit

PL2	'1'	STRING	-	-	1	1	ID Control : V_Exit_Deep_Sleep_mode_and_WakeUp	

ESPrtk Respond

Payload Index		Type value	Min Value	Max Value	Min Length	Max Length	Describe		Unit
PL2	'1'						ID Control : V_Exit_Deep_Sleep_mode_and_WakeUp		
	PL3	INTEGER	-1	1	-	-	Status Control		
							-1	NACK: Invalid input /wrong format / out of range …?	
							0	- Timeout : Not see confirm message from HOST	
							1	OK: Exit Deep Sleep mode and WakeUp success	
	PL4	STRING	-	-	-	-	Describe status respond control as text		

Example

Message :	
#1 HOST Send	$ESP_OK|6|T|V|1|*6426
#1 ACK Respond	Please send $ESP_OK|6|T|V|1|*6426 again to confirm wake up !
#2 HOST Send again to confirm wake up	$ESP_OK|6|T|V|1|*6426
#2 ACK Respond	$ESP_OK|49|T|V|0|1|Exit Deep Sleep mode and wake up success|*026C
NACK Respond	
Other NACK	$ESP_ER|… or $ESS_ER|… or $AES_ER|…

WEB Configure mode

Besides using UART_Configure to configure ESPrtk.
Users can perform ESPrtk configuration through a WEB interface (UI) with a web browser running on a computer or mobile device.

Start WEB Configure mode.

There are two ways to start WEB Configure mode on ESPrtk.

- Use ESPrtk command control
- Use the CWB button

Use ESPrtk command to start WEB configure

Use ESPrtk command control	
Step 0	ESPrtk after supplying power.
Step 2	HOST send command : **$ESP_OK|5|N|_||*451F**
Done	ESPrtk will start go to WEB Configure mode. The CONNECT_LED blink fast.
Success	The CONNECT_LED will light continuously . ESPrtk respond on TX_UART0: **$ESP_OK|5|N|_||*451F**

Use the CWB button to start WEB configure

Method 1: Use a combination of RESTART and CWB buttons.	
Step 0	ESPrtk after supplying power.
Step 1	Press and hold the CWB button. (set GPIO to Low level)
Step 2	Press the RESET button (then release)
Done	Continue to hold CWB in 0.5 second -> ESPrtk will start go to WEB Configure mode. The CONNECT_LED blink fast.
Success	The CONNECT_LED will light continuously . ESPrtk respond on TX_UART0 : **$ESP_OK|5|N|_||*451F**
Method 2: Use only the CWB button	
Step 0	ESPrtk after supplying power.
Done	Press and hold the CWB button (set GPIO to Low level) for 3 seconds -> ESPrtk will start go to WEB Configure mode. The CONNECT_LED blink fast.
Success	The CONNECT_LED will light continuously .

	ESPrtk respond on TX_UART0: **$ESP_OK|5|N|_||*451F**

TIP :
After ESPrtk enters WEB Configure mode successfully, pressing the CWB button will not work in this mode.
When using the CWB button to launch WEB Configure, both ways you need to hold it down, so if pressing and holding the button makes it difficult for you, you may need to use a switch (parallel connection with the CWB button).

Access WEB UI and use

Now ESPrtk will start the WiFi hotspot (**AccessPoint**) named: **ESPrtk : xxxxx**. With xxxxx is the ID of ESPrtk device. (ESPrtk_ID).
When the boot process is complete, the connected CONNECT_LED will light continuously.
You can now connect to the WiFi hotspot , because the station is public (open) so you can connect without a password.
After connecting, open any browser on your device, and access **http://192.168.4.1/profile** to enter **Profile** page.
At the profile page, click on the “Login/Logout” tab, enter your username and password and then click “Login”.
On ESPrtk 3.0.0 , the default username/password is **admin** and **abc123ABC**
If ESPrtk does not have an account yet, click on “**Create new profile**” with your username/password chosen by user.

	Tab	Access address
1	**Homepage**	http://192.168.4.1
2	**Ublox Tab**	http://192.168.4.1/ublox
3	**Navspark Tab**	http://192.168.4.1/navspark
4	**GNSS Viewer Tab**	http://192.168.4.1/gnssviewer
5	**Profile Tab**	http://192.168.4.1/profile
6	**About Tab**	http://192.168.4.1/about
7	**System Tab**	http://192.168.4.1/system

Homepage

Introduce

WEB Configure homepage.

Click on the middle of the screen to go to the Profile configuration page.

Ublox Tab

Introduce

This function allows users to configure Ublox module in the same way as Ublox Center software. Just connect the UART1 port (ESPrtk) to the UART-Configure (Ublox module), select the appropriate baudrate and start the configuration.

This function uses the latest Ublox Center API and protocol and supports configuration on all of Ublox's latest product lines such as F9P-M8P-LEA -NEO.

Hardware connect

Type	ESPrtk pin name	ZED-F9P	M8P – LEA -NEO	Other Ublox modules
UART	TX_UART1	RX_UART_1	RX_UART_1	RX UART-config
UART	RX_UART1	TX_UART_1	TX_UART_1	TX UART-config
UART	TX_UART2	RX_UART_2		RX UART-config
UART	RX_UART2	TX_UART_2		TX UART-config

Support function

Communicate	Support
UART1 / UART2 Switch + Multi baudrate + Auto scan baudrate Ublox device	●
Automatically detect errors of input data.	●

	Ublox Configure function	Support
0	**ANT (Antenna Settings)**	●
1	**BATCH (Batch mode output)**	●
2	**CFG (Configuration)**	●
3	**DAT (Datum)**	●
4	**DGNNS (Differential GNSS configure)**	●
5	**DOSC (Disciplined Oscillator)**	●
6	**EKF (EKF Settings)**	○
7	**ESFGWT (Gyro + Wheeltick)**	○
8	**ESRC (External Source Config)**	●

9	**FXN (Fix Now Mode)**	○
10	**GEOFENCE (Geofence Config)**	●
11	**GNSS (GNSS Config)**	●
12	**HNR (High Nav Rate)**	●
13	**INF (Inf Message)**	●
14	**ITFM (Jamming/Interference Monitor)**	●
15	**LOGFIL (Log Settings)**	●
16	**MSG (Messages)**	●
17	**NAV5 (Navigation 5)**	●
18	**NAVX5 (Navigation Expert 5)**	●
19	**NMEA (NMEA Protocol)**	●
20	**ODO (Odometer/Low-Speed COG filter)**	●
21	**PM (Power Management)**	○
22	**PM2 (Extended Power Management)**	●
23	**PMS (Power Management Setup)**	●
24	**PRT (Ports)**	●
25	**PWR (Power)**	●
26	**RATE (Rates)**	●
27	**RINV (Remote Inventory)**	●
28	**RST (Reset)**	●
29	**RXM (Receiver Manager)**	●
30	**SBAS (SBAS Settings)**	●
31	**SLAS (SLAS Settings)**	●
32	**SMGR (Sync Manager Config)**	●
33	**TMODE (Time Mode)**	○
34	**TMODE2 (Time Mode2)**	●
35	**TMODE3 (Time Mode 3)**	●
36	**TP (Timepulse)**	●
37	**TP5 (Timepulse 5)**	●
38	**TXSLOT (Tx Time Slots)**	●
39	**USB (Universal Serial Bus)**	●
40	**VALDEL (New Configuration)**	○
41	**VALGET (New Configuration)**	○
42	**VALSET (New Configuration)**	○

43	**ESPrtk - Hex configure Ublox tool**	●

To use

Connect UART port as table above.

- 1- First, start connecting to the module with a baudrate.
- 2-Use "Autoscan Baudrate" to scan if dont know baudrate.
- 3-Select configure function , select yourconfigure item , press SEND configure .

TIP :

- Wait a moment after changing the baudrate.
- Do not click too quickly (after configure, you need wait a moment).
- Items marked with '*' are not supported yet.
- Scroll the Configure tab to see the full component.

ESPrtk - Hex configure Ublox tool

This function allows users to configure Ublox modules with Hex code.

(These codes are the PAYLOAD part in the communication frame to configure Ublox's module, can find the communication frame at the configuration documents).

ESPrtk also supports syntax errors filter , error alerts, data dumps, auto-sending …

The input field has the format:

Input field format
<Configuration Name> – <Hex code> <Enter>

For example:

CFG-MSG – 06 01 08 00 0A 06 00 00 00 00 00 00
CFG-MSG – 06 01 08 00 0A 07 00 00 00 00 00 00
CFG-MSG – 06 01 08 00 0A 21 00 00 00 00 00 00

Check input:

- Configuration name and Hex code are separated by dashes: '-'.
- Each hex value must be separated by a space.
- The configuration commands must be separated by new lines (enter).
- A Byte Hex must contain at least 1 characters (and no more than 2 characters).
- Irrespective of uppercase and lowercase letters.

For example, all four ways below are valid and have the same output:

- Hex 1 = 01 02 0F 07 00 0D
- Hex 2 = 1 2 F 7 0 D
- Hex 3 = 1 2 f 7 0 d
- Hex 4 = 01 2 f 07 0 d

All of the four Hex codes above have a unique output:

- Hex = 01 02 0F 07 00 0D

Navspark Tab

Introduce

Hardware connect

Type	ESPrtk pin name	NS-HP-BD NS-HP-GL NS-HP-GN PX1120R	PX1122R	NS-HP-xx	Other Navspark modules
UART	TX_UART1	RX_UART_1	RX_UART_1	RX_UART_1	RX UART-config
UART	RX_UART1	TX_UART_1	TX_UART_1	TX_UART_1	TX UART-config
UART	TX_UART2	RX_UART_2	RX_UART_2	RX_UART_2	RX UART-config
UART	RX_UART2	TX_UART_2		TX_UART_2	TX UART-config

Support function

Communicate	Support
UART1 / UART2 Switch + Multi baudrate + Auto scan baudrate Navspark device	●
Automatically detect errors of input data.	●

	Binary - Configure function	Support	Binary - Query function	Support
0	Configure Serial Port	●	Query Sofware Version	●
1	Configure Message Type	●	Query CRC Checksum Version	●
2	Configure Pinning Parameters	●	Query SHA1 String	◌
3	Configure NMEA Message Interval	●	Query NMEA Message Interval	●
4	Configure Binary Message Interval	●	(blank)	
5	Configure Position Update Rate	●	Query Position Update Rate	●
6	Configure Position Pinning	●	Query Position Pining	●
7	Configure Power Mode	●	Query Power Mode	●
8	Configure DOP Mask	●	Query DOP Mask	●
9	Configure Elevation and CNR Mask	●	Query Elevation and CNR Mask	●
10	(blank)	●	Query GLONASS K-Number, Slot, CNR	◌
11	Configure NMEA Talker ID	●	Query NMEA Talker ID	●
12	Configure NMEA Output Comport	◌		
	Venus 8 - Configure function	**Support**	**Venus 8 - Query function**	**Support**
0	Configure SBAS	●	Query SBAS	●
1	Configure SBAS Advance	◌	Query SBAS Advance	◌
2	Configure SAEE	●	Query SAEE	●
3	Configure QZSS	●	Query QZSS	●
4	Configure DGPS	◌	(blank)	
5	Configure Interference Detect Control	●	Query Interference Detect Control	●
6	Configure Parameter Search Engine Number	●	Query Parameter Search Engine Number	●
7	Configure Position Fix Navigation	●	Query Position Fix Navigation	●

	Mask		Mask	
8	Configure Navigation Mode	●	Query Navigation Mode	●
9	Configure GNSS Constellation Type	●	Query GNSS Constellation Type	●
10	Configure GPS/UTC Leap Seconds	●	Query GPS Time	●
11	Configure Parameter Search Engine Sleep Criteria	○	Query Parameter Search Engine Sleep Criteria	○
12	Configure Datum Index	●	Query Datum Index	●
13	Configure Kernel Very Low Speed		Query Kernel Very Low Speed	○
14	Configure PSTI Interval	●	Query PSTI 004 Interval	●
15	Configure PSTI Interval	●	Query PSTI 007 Interval	●
16	Configure PSTI Interval	●	Query PSTI 020 Interval	●
17	Configure PSTI Interval	●	Query PSTI 030 Interval	●
18	Configure PSTI Interval	●	Query PSTI 032 Interval	●
19	Configure PSTI Interval	●	Query PSTI 033 Interval	●
20	Configure PSTI Interval	●	Query PSTI 060 Interval	●
21	Configure PSTI Interval	●	Query PSTI 063 Interval	●
22	Configure PSTI Interval	●	Query PSTI 065 Interval	●
23	Configure PSTI Interval	●	Query PSTI 067 Interval	●
24	Configure PSTI Interval	●	Query PSTI 068 Interval	●
35	Configure PSTI Interval	●	Query PSTI 070 Interval	●
36	Configure Custom PSTI from 000 to 255 Interval	●	Query Custom PSTI Interval	○
	Phoenix - Configure function	**Support**	**Phoenix - Query function**	**Support**
0	Config Phoenix Tag	○	Query Phoenix Extended ID	○
1	Configure Custom NMEA string Interval	●	Query Custom NMEA String Interval	○
2	Configure NMEA string Interval	●	Query GGA NMEA String Interval	●
3	Configure GNS NMEA String Interval	●	Query GNS NMEA String Interval	●
4	Configure GSA NMEA String Interval	●	Query GSA NMEA String Interval	●
5	Configure GSV NMEA String Interval	●	Query GSV NMEA String Interval	●
6	Configure GLL NMEA String	●	Query GLL NMEA String	●

	Interval		**Interval**	
7	**Configure RMC NMEA String Interval**	●	**Query RMC NMEA String Interval**	●
8	**Configure VTG NMEA String Interval**	●	**Query VTG NMEA String Interval**	●
9	**Configure ZDA NMEA String Interval**	●	**Query ZDA NMEA String Interval**	●
10	**Configure DTM NMEA String Interval**	●	**Query DTM NMEA String Interval**	●
11	**Configure GBS NMEA String Interval**	●	**Query GBS NMEA String Interval**	●
12	**Configure GRS NMEA String Interval**	●	**Query GRS NMEA String Interval**	●
13	**Configure GST NMEA String Interval**	●	**Query GST NMEA String Interval**	●
14	**Configure THS NMEA String Interval**	●	**Query THS NMEA String Interval**	●
15	**Configure HDT NMEA String Interval**	●	**Query HDT NMEA String Interval**	●
	RAW - Configure function	**Support**	**RAW - Query function**	**Support**
0	**Configure Base Station**	●	**Query Base Position**	◌
1	**Configure Binary Measurement Data Out**	●	**Query Binary Measurement Data Out**	●
2	**Configure RTCM Measurement Data Out**	●	**Query RTCM Measurement Data Out**	●
	RTK - Configure function	**Support**	**RTK-Query function**	**Support**
0	**Configure RTK Reference Static Started Position**	●	**Query RTK Mode**	●
1	**Configure RTK Mode**	●	**Query RTK And Operational Function**	●
2	**Configure RTK Mode And Operational Function**	●	**Query RTK Mode And Operational Function**	●
3	**Configure RTK Slave Serial Port Baud Rate**	●	**Query RTK Slave Serial Port Baud Rate**	●
4	**Configure Precisely Kinematic Base Serial Port Baud Rate**	●	**Query RTK Precisely Kinematic Base Serial Port Baud Rate**	●
5	**Configure RTK GLONASS Carrier-Phase Inter-Frequency Bias**	◌		
	1PPS Timing- Configure function	**Support**	**1PPS Timing - Query function**	**Support**

0	**Configure 1PPS Timing**	●	**Query 1PPS Timing**	●
1	**Configure 1PPS Cable Delay**	●	**Query Cable Delay**	●
2	**Configure 1PPS Pulse Width**	●	**Query 1PPS Pulse Width**	●
3	**Configure 1PPS Output Mode**	●	**Query 1PPS Output Mode**	●
4	**Configure 1PPS Frequency Output**	●	**Query 1PPS Frequency Output**	●
	Ephemeris - Configure function	**Support**	**Ephemeris - Query function**	**Support**
0	**Configure GPS Ephemeris**	○	**Query GPS Ephemeris**	○
1	**Configure GLONASS Ephemeris**	○	**Query GLONASS Ephemeris**	○
2	**Configure Beidou Ephemeris**	○	**Query Beidou Ephemeris**	○
3	**Configure GPS Almanac**	○	**Query GPS Almanac**	○
4	**Configure GLONASS Almanac**	○	**Query GLONASS Almanac**	○
5	**Configure Beidou Almanac**	○	**Query Beidou Almanac**	○
6	**Configure GLONASS Time Corrections**	○	**Query GLONASS Time Corrections**	○
	Pro Configure - Configure function	**Support**	**Pro Configure - Query function**	**Support**
0	**Hex Payload Configure**	●		

To use

Connect UART port as table above.

- 1- First, start connecting to the module with a baudrate.
- 2-Use "Autoscan Baudrate" to scan if dont know baudrate.
- - To check the response from moudle Navspark, you can send "Query Software Version".
- 3-Select configure function , select your configure item , press SEND configure .

TIP :

- Wait a moment after changing the baudrate.
- Do not click too quickly (after configure, you need wait a moment).
- Items marked with '*' are not supported yet.
- Scroll the Configure tab to see the full component.
- When you see the error [Module respond NACK (just not accept, please choose another option)], it means that the connection is successful and correct baudrate, but the configure value is invalid, you need to check the valid values of that module.

ESPrtk - Hex configure Navspark tool

This function allows users to configure Navspark modules with Hex code.

(These codes are the PAYLOAD part in the communication frame to configure Navspark's module, can find the communication frame at the configuration documents).

ESPrtk also supports syntax errors filter , error alerts, data dumps, auto-sending …

The input field has the format:

Input field format
<Configuration Name> – <Hex code> <Enter>

For example:

[NMEA_Message_Interval] - 08 01 01 01 01 01 01 00 00

[NMEA_Message_Interval] - 08 01 01 01 01 01 01 00 00

Check input:

- Configuration name and Hex code are separated by dashes: '-'.
- Each hex value must be separated by a space.
- The configuration commands must be separated by new lines (enter).
- A Byte Hex must contain at least 1 characters (and no more than 2 characters).
- Irrespective of uppercase and lowercase letters.

For example, all four ways below are valid and have the same output:

- Hex 1 = 01 02 0F 07 00 0D
- Hex 2 = 1 2 F 7 0 D
- Hex 3 = 1 2 f 7 0 d
- Hex 4 = 01 2 f 07 0 d

All of the four Hex codes above have a unique output:

- Hex = 01 02 0F 07 00 0D

GNSS Viewer Tab

Introduce

Hardware connect

Type	ESPrtk IO name	IMU -MPU9250	Ublox module	Navspark module	Other GNSS modules
I^2C	SCL_I2C	SCL I^2C	-	-	-
I^2C	SDA_I2C	SDA I^2C	-	-	-
UART	TX_UART1	-	RX_UART_1	RX_UART_1	RX UART
UART	RX_UART1	-	TX_UART_1	TX_UART_1	TX UART – NMEA/RTCM Output
UART	TX_UART2	-	RX_UART_2	RX_UART_2	RX UART
UART	RX_UART2	-	TX_UART_2	TX_UART_2	TX UART – NMEA/RTCM Output

Support function

Hardware communicate	Support
UART1 / UART2 Switch + Multi baudrate	●

	GNSS Viewer tool	IMU Viewer tool
1	**Information Table**	**3D Model Xbox view+ Bias**
2	**GPS Sattelite View**	**3D Model F117 view+ Bias**
3	**BEIDOU Sattelite Signal View**	**Euler Angle graph view**
4	**GLONASS Sattelite View**	**Gyroscope graph view**
5	**GPS Sattelite Sky View**	**Accelerometer graph view**
6	**BEIDOU Sattelite Sky View**	**Euler Angle chart view**
7	**GLONASS Sattelite Sky View**	**Gyroscope chart view**
8	**Position Viewer Tool**	**Accelerometer chart view**

To use

Connect the RX-UART1 port to the TX output of the GNSS device.

- 1- First, Select connection baudrate and screen update frequency then press connect.

- Wait in 5 second until NMEA/RTCM information show/draw on page.

TIP :

If the input is NMEA, the following messages will be parse: RMC-GLL-GGA – GSA-ZDA-VTG-GSV. If the input is RTCM, ESPrtk 3.0.0 currently can only detect RTCM 3.x and show at "Message Type Parse " box but does not really resolve it because it is not fully supported yet.

Profile Tab

Introduce

Hardware connect

Type	ESPrtk IO name	IMU -MPU9250	SD Card
I^2C	SCL_I2C	SCL I^2C	-
I^2C	SDA_I2C	SDA I^2C	-
SPI	MOSI_SD_Card	-	MOSI
SPI	MISO_SD_Card	-	MISO
SPI	SCK_SD_Card	-	SPI Clock
SPI	CS_SD_Card	-	Chip Select

Support function

Communicate	Support
Automatically detect errors of input data.	●

	Function tab	UI Write configure for	UI Auto Read configure	Other UI Tools
0	**Create new profile**			
1	**Login / Logout**			
2	**Your profile**	**YOUR_PROFILE**	●	
3	**Action Planning**	**ACTION_PLANNING**	●	
4	**Internet - RTK Online**	**INTNERNET**	●	
5	**Radio - RTK OFFline**	**RADIO_RTK**	●	
6	**MQTT RTK**	**MQTT_RTK**	●	

7	**NTRIP Client**	**NTRIP_CLIENT**	●	**Mountpoint Table Insert**
8	**NTRIP Master**	**NTRIP_MASTER**	●	
9	**UART Configure**	**UART_CONFIGURE**	●	**Command Tester Tool**
10	**Bluetooth**	**BLUETOOTH**	●	
11	**Display viewer**	**DISPLAY_VIEWER**	●	
12	**Log File-Events**	**EVENT_LOG**	●	**Event log Decoder**
13	**Simple Debug**	**SIMPLE_DEBUG**	●	**Convert ID _Error to Text**
14	**SD Card Logger**	**SD_CARD**	●	
15	**IMU-MPU9250**	**IMU_MPU9250**	●	**Calibration IMU Helper**
16	**GSM 2G-3G-4G-LTE**	**GSM_CELLULAR**	●	**GSM - ATCode Edit + Decoder**
17	**Stream Manager Remove+Insert+Add**	**STREAM_MANAGER**	●	**STREAM - AT Code Edit + Decoder**
18	**Position+Antenna**	**POSITION_ANTENNA**	●	

NTRIP Client - Mountpoint Table Insert tool.

This function automatically updates and provides a mountpoint list with basic information of Basestation in the form of resolved data tables. Users only need to select from the list, the mountpoint name will be filled automatically. The function will also automatically detect and suggest entering additional information such as Password, position, ... if that NTRIP service requires.

The sourcetable data storage size on ESPrtk is up to 200 Kbytes. (If the data is too large, some data of the end will be remove).

UART Configure – Command Tester tool.

One of the special features available on version 3.0.0.

This function simulates a virtual UART port and an INUPUT / OUTPUT window similar to the Serial Terminal software running on a PC.

This function is used to practice or configure ESPrtk directly using the ESPrtk command (those used in UART Configure).

Users can also use this function to read all configuration information or configure all in one click.

The support:

- Automatically detect message frame errors and propose corrections.

- Automatic separation, sending and counting of successful message frames.

- Supports all **Command Read Configure ESPrtk .** (see detail ID Commands above)

- Support all **Command Write Configure ESPrtk**. (see detail ID Commands above)

- Support encryption framework $ ESS_xx and $ AES_xx (BITWISE and AES256).

- Supports some basic control commands.

PL0	Type Command	WEB Configure Mode Support ?- Command Tester Tool
'N'	Start WEB Configure mode	●
'O'	Start UART Configure mode	●
'X'	Start Main Action mode	●
'P'	PING to ESPrtk	●
'Q'	Quit UART Configure mode and jump to Main Action mode	◌
'R'	Read data configure on ESPrtk	●
'W'	Write configure to ESPrtk	●
'E'	Export configure as example	●
'T'	Hardware control	◌
'T'	Hardware control (Control type PRINT output on TX_UART0)	●
'T'	Hardware control (Control GPIO ESPrtk)	●
'R'	Request ERROR_SYSTEM	●

Log File-Events - Event log Decoder tool.

The best way to see the status of ESPrtk is to collect TX-UART0 output data, the purpose of this is to find the error and fix it.
However, this requires a continuous connection from ESPrtk to the terminal display device, and will make it difficult for users without the necessary tools, or distance obstacles.
Event Log will overcome that disadvantage.
When the Eventlog is activated, during operation, data about the operation, error status, .. will be recorded in a file named Event_Log.bin and stored forever until the user delete it.

During operation, users can stop the program whenever they feel that collecting data for the Event_Log file is enough, then they can check the Event_Log file to see what happened on ESPrtk, (or how ESPrtk works until now).

Users can also download this file to store or share with everyone.
The Event_Log file will be limited by the maximum allowable size.
Users can select data collection targets. For debugging should activate all.

If Event_Log is enabled, it starts to record the status from the time ESPrtk is restarted (power on or press the Reset button) until the user turns off the power (or press the Reset button).

Now, in ESPrtk, there exists an Event_Log.bin file, users can open Webconfigure, go back to the Event Log tab and click on the READ File button to extract the file, need to wait for a while (60s longest) for a series of Information will be displayed below.

The user can then download this file or need to delete the file by clicking Clear File to allow a new recording process at the next operation.

Note:
Activating Event_Log will cause memory resources on ESPrtk to be used (depending on the maximum size allowed from 5Kbyte to 50Kbyte).
The process of extracting Event_Log.bin file sometimes takes a lot of time because of the large size.
The Event_Log.bin file will not contain sensitive installation information in the Profile such as passwords, IDs, host servers, …
The file Event_Log.bin is not encrypted protected, it is just a zip file, it is open source and can be read normally from the mapping table to extract it. Its data is the result of the active log collection process on ESPrtk and is not involved in ESPrtk's operational configuration.
You can share this file with everyone.

Simple Debug - Convert ID _Error to Text tool.

When you collect the ID of the error on ESPrtk (ID number or color seen on Neopixel), you can decode it using this simple tool to see what error has occurred.

After logging in, open the Simple Debug tab, enter the error ID or select the neopixel strip color in order (1-2-3-4).

Press "See error", the page will print an error in the form of text and suggest how to fix it.

This is just the simplest debug tool, sometimes it will help you when you can't contact anyone to get help.

IMU-MPU9250 - Calibration IMU Helper tool.

This tool allows users to collect raw and calibrated magnetometer data of the IMU sensor to assess the quality of the raw or calibrated magnetometer using the conversion matrix.

To use this function, connect the MPU9250 sensor to the I2C port of ESPrtk.

The support:

- Export raw and calibrated data simultaneously on all 3 X-Y-Z axes.
- Support to configure the output data format according to the purpose of the user.
- Display points and interact with 3D graphics.
- Support compressing data file into .txt or .svc file for download.
- Automatic stop by preset size limit.

About Tab

Introduce

Support function

Other UI Tools
Request ESPrtk information

System Tab

Introduce

Support function

	Function tab	UI Write configure for	UI Auto Read	Other UI Tools

			configure	
0	**ESPrtk Custom Pin Map**	**GPIO_PIN_MAP**	●	**Input GPIO pin map as JSON string**
1	**Setting system**	**SYSTEM_Other_Mode**	●	

Input GPIO pin map as JSON string tool.

This is a special feature and it is hidden when accessed in the Web Config.
To be able to access this page, after the login Profile, type 192.168.4.1/system to enter System page.

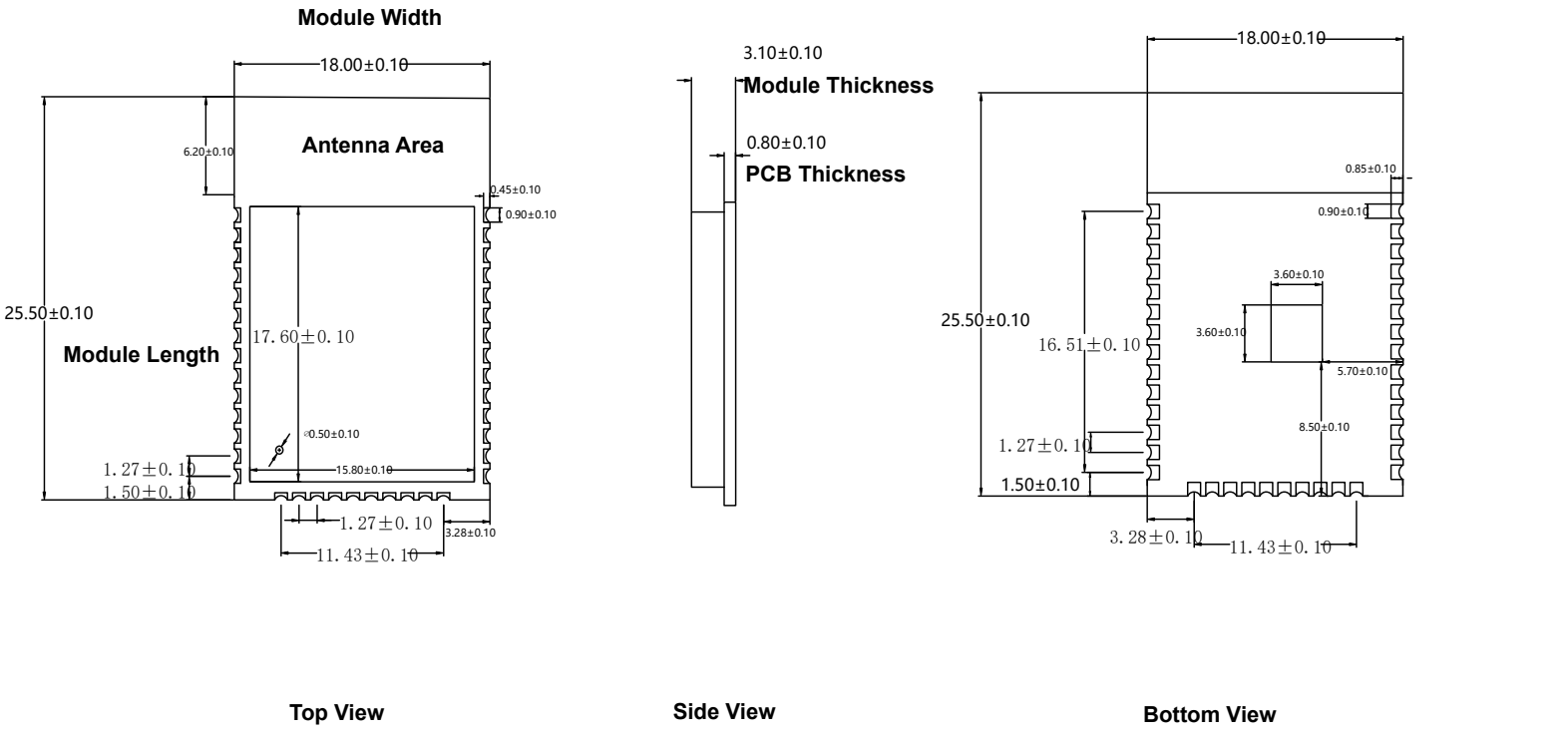
This page showing with a black background interface , users can drag and drop ESP32 pins into the frame corresponding to ESPrtk. Click “Save” button to save the configuration.
Or user can enter the JSON string into the text field and click “Save as JSON” to save the configuration.
After that, ESPrtk will check the logic, if there is no error, it will print “Config GPIO save success” in the Submit status box.

At the same time, it export a JSON string, clicks “Copy output” to save it in Clipboard as text and paste it anywhere.
Configure data will store in FLash in ESP32 .
After restarting , GPIO configuration information will be printed on the Serial port (TX_UART0 baud 230400) and load this configure to all function before main program start.

Physical Dimensions

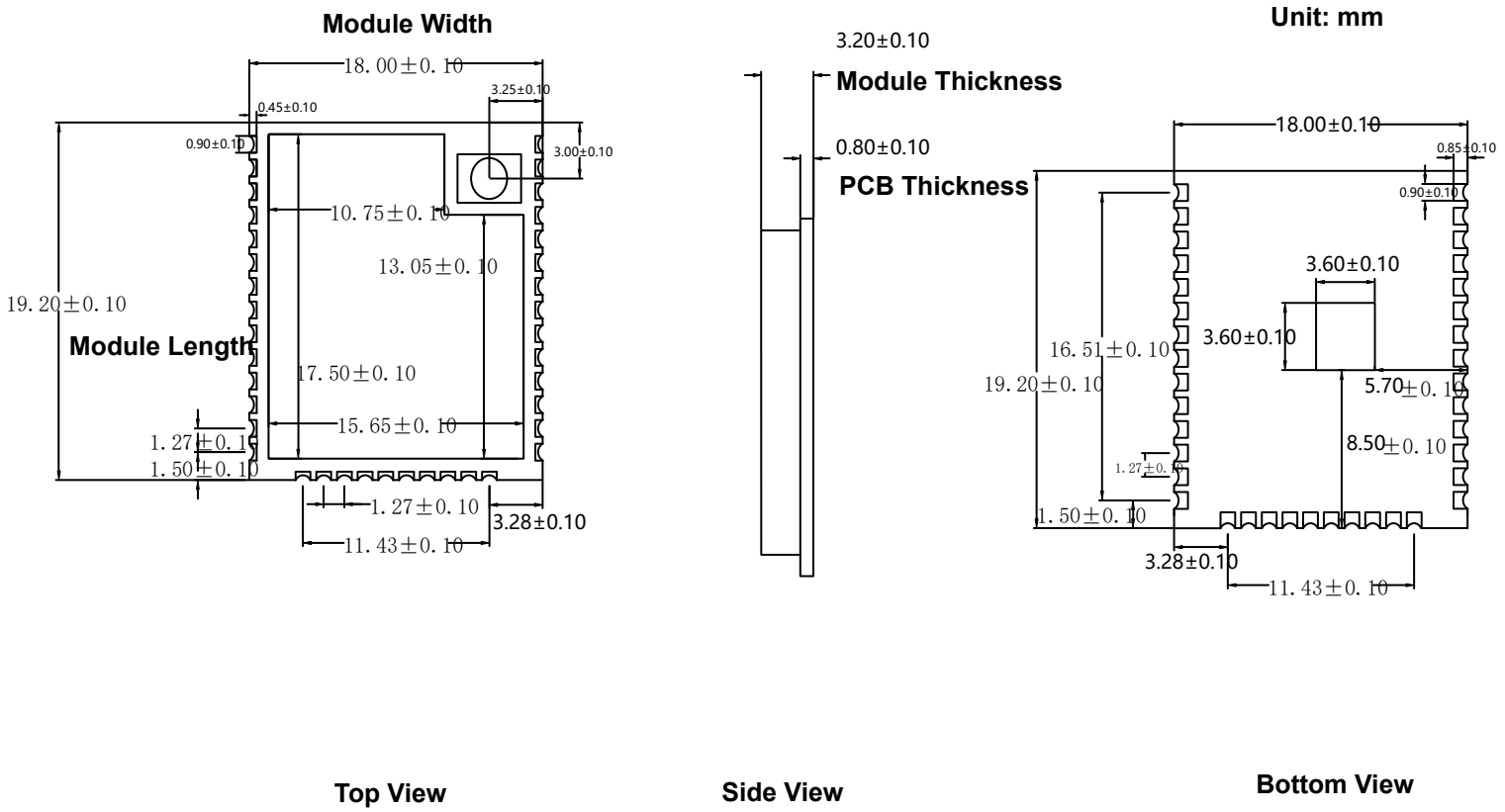
ESP32-WROOM-32D DIMENSIONS

Unit: mm

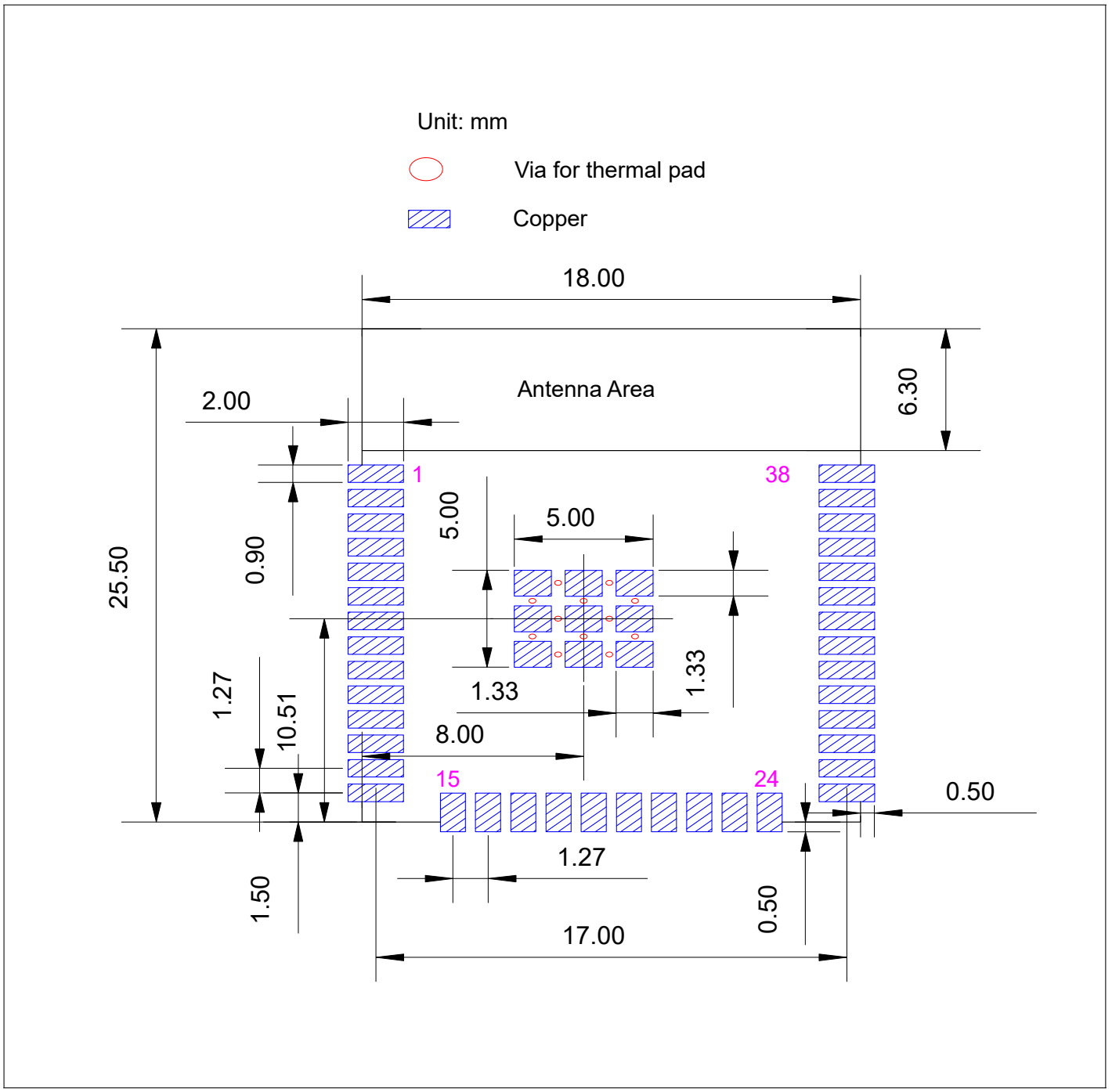


Physical Dimensions of ESP32-WROOM-32D/E

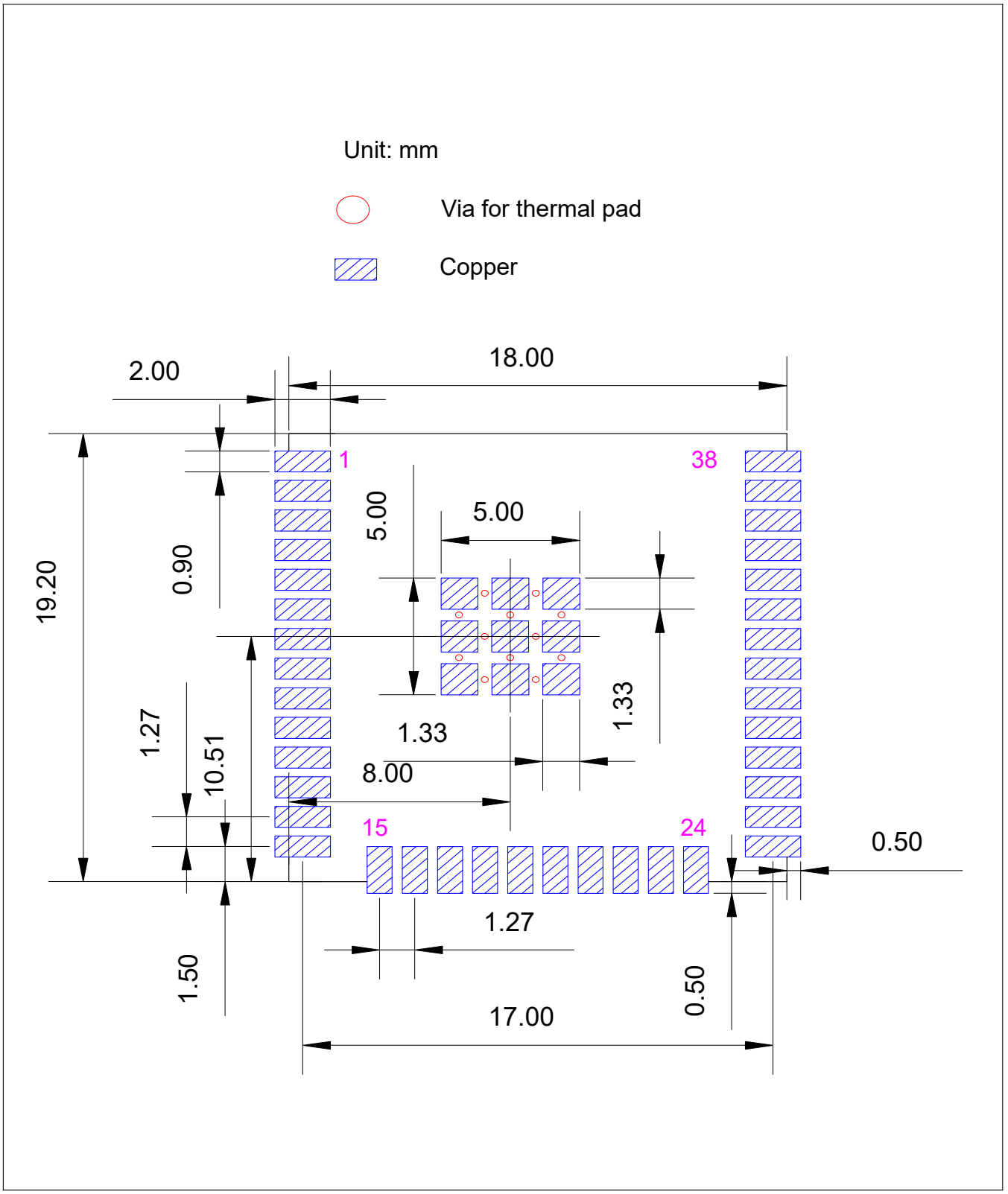
ESP32-WROOM-32U DIMENSIONS



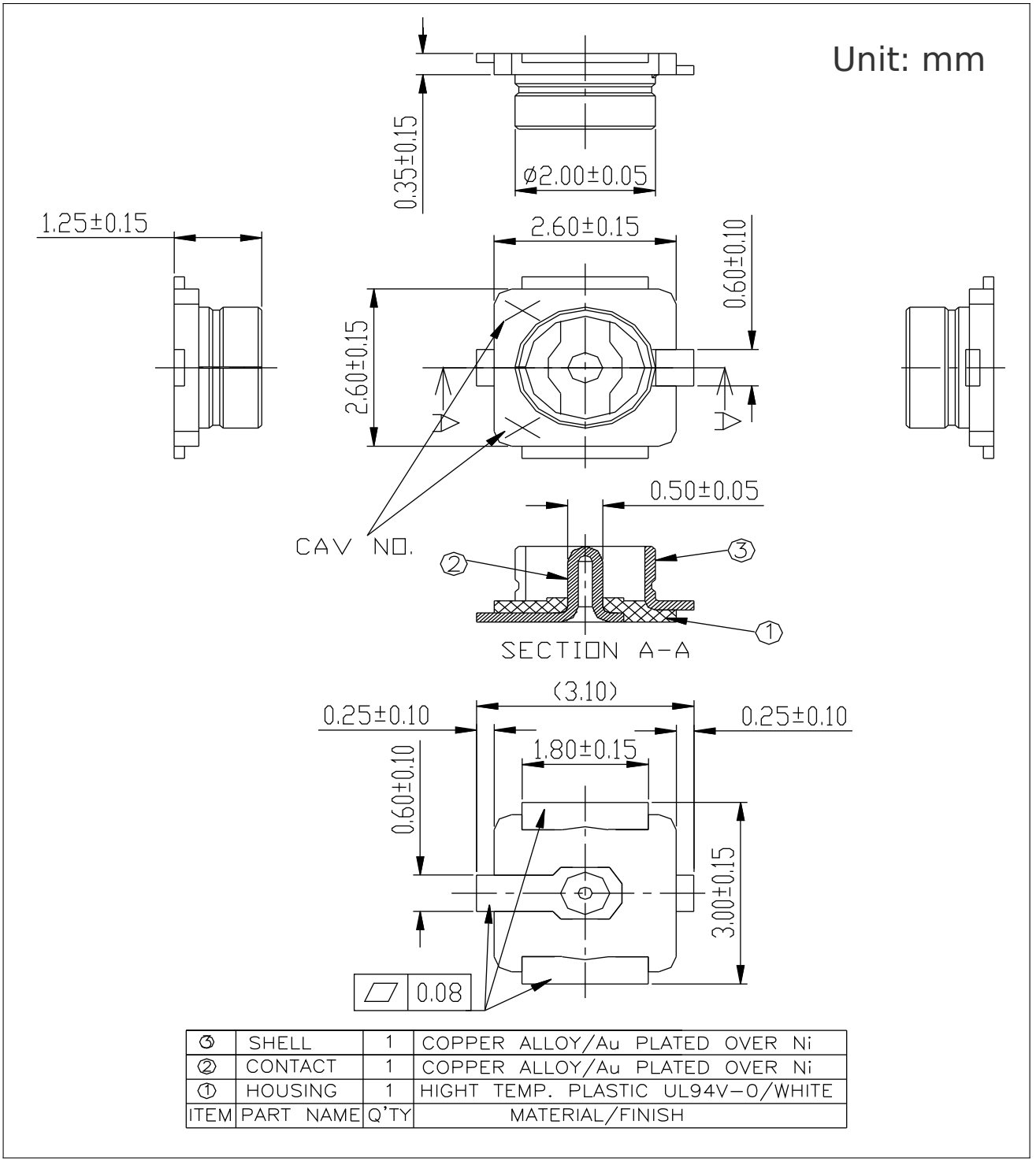
Physical Dimensions of ESP32-WROOM-32U/UE



Recommended PCB Land Pattern of ESP32-WROOM-32D/E



Recommended PCB Land Pattern of ESP32-WROOM-32U/UE



ITEM	PART NAME	Q'TY	MATERIAL/FINISH
③	SHELL	1	COPPER ALLOY/Au PLATED OVER Ni
②	CONTACT	1	COPPER ALLOY/Au PLATED OVER Ni
①	HOUSING	1	HIGHT TEMP. PLASTIC UL94V-0/WHITE

ESP32-WROOM-32U/UE U.FL Dimensions