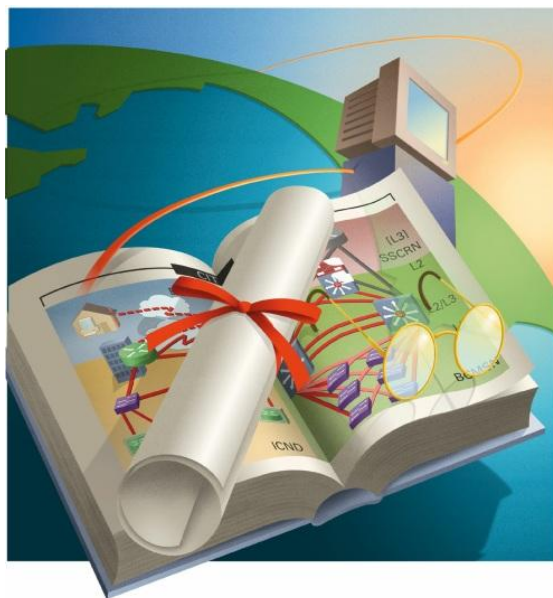


iEdge 4.0 White Paper



Polysense Technologies Inc.
Oct.18th 2022

Preface

Ten years ago, the IOT was still a very small concept, but its prospects were widely optimistic. All major research and consulting institutions were very optimistic about the IOT industry, predicting that the number of IOT connections would reach 20 billion, 50 billion, 100 billion or more in the future.

Under the temptation of the huge market cake, many players have invested in the industrial army of the IOT, which has greatly promoted the development of the IOT industry. Up to now, the IOT industry has made a series of achievements.

1. First of all, there is the vigorous development of technology. The IOT industry can be divided into four layers: the perception layer, the transmission layer, the platform layer and the application layer. Each layer has different technologies. Ten years ago, or even five years ago, there were not many technologies available for IOT products, which were basically based on the old technology "new wine in old bottles" tens of years ago. As a result, either the performance did not meet expectations or the cost performance was too low. In recent years, IOT technologies at all levels have been greatly developed, such as smart sensors, RFID products with better performance, 5G, NB IoT, Cat 1. The emergence and evolution of new technologies such as LoRa, Bluetooth 5. X, Wi Fi 6, edge computing, AI algorithm and so on make IoT products more suitable for market demand.

2. Secondly, it is the application level. Compared with earlier years, the public's perception of the IOT is "only hearing its voice, not its use". At present, the application of the IOT has penetrated into all aspects of people's production and life, such as taxi taking, shopping, bike sharing, smart door locks, smart speakers, security cameras, and so on. According to our research, at present, the shipment of various RFID products has reached more than 20 billion annually, the shipment of Wi Fi, Bluetooth, ZigBee and other connection technologies has reached the level of billions every year, and the shipment of NB IoT, LoRa and other technologies has reached more than 100 million every year. These huge connection figures can be transformed into rich and diverse applications.

3. Finally, at the level of industrial richness, compared with the concept of the IOT as a niche in the early years, in recent years, the IOT has been favored by Huawei, Alibaba, Tencent, three major operators, Xiaomi and other giants. Big factories all regard IoT as a "new track" leading to the future. In addition, more supporting manufacturers and entrepreneurs have entered the field of the IOT, The variety and number of players in the whole industry have been very rich. Although the IOT industry has made a series of achievements, there are still many problems in this industry. On the one hand, fragmentation is serious. There are multiple levels of fragmentation, including fragmentation of technology, fragmentation of applications and fragmentation of players.

1. The fragmentation of technology means that there are many IoT technology schools, and there are often a variety of technology options to address the needs of a project, which results in that enterprises need to accumulate a variety of IoT technologies to better match the market demand.

2. The fragmentation of applications means that the application projects of the IOT are diverse, such as industry, city, transportation, logistics, parks, agriculture, medical care, etc. Even based on these sub industries, more abundant gap fields will be extended. Each field has different industry rules and different industrial chains. Therefore, for the IOT enterprises, a lot of energy needs to be invested to study the applications of various industries.

3. The fragmentation of players means that there are many types of players in the IOT enterprises. Because the concept of the IOT is vague and there is no unified standard, a large number of traditional enterprises can enter the IOT based on their own understanding, which has resulted in a situation where products and solutions are mixed in the IOT market and users are difficult to distinguish.

Of course, intelligence is an irreversible trend of the whole society, and there is no doubt about the application prospect of the IOT in the future. However, at the current stage, the whole industry is still in a "chaotic stage". How will it evolve in the future requires the choice of the market to give the answer.

1. Overview	6
1.1 Background	6
1.1.1 Company introduction	6
1.1.2 Product development background	6
1.2 Product positioning	7
2. Product attributes	8
2.1 Product Matrix	8
2.2 Product iteration	10
2.3 Product highlights	11
3. System Architecture	11
3.1 System framework	11
3.1.1 Application layer	12
3.1.2 Middle layer	12
3.1.3 OS layer	12
3.1.4 Kernel layer	13
3.1.5 Device layer	13
3.2 Structure and appearance	13
3.2.1 Reliable structural design	13
3.2.2 Universal design	20
3.2.3 Installation Guide	20
3.3 Hardware	22
3.3.1 Technical specifications	22
3.3.1.1 MCU	22
3.3.1.2 Universal main board	22
3.3.1.3 Communication board	25
3.3.1.4 Batterypower board	28
3.3.1.5 Antenna	28
3.3.1.6 Hardware interface	28
3.3.2 Scalability	30
3.3.3 Certification report	30
3.4 Software	30
3.4.1 System architecture	31
3.4.1.1 Architecture diagram	31
3.4.1.2 System characteristics	31
3.4.2 iEdge OS kernel	32
3.4.2.1 Boot OS	32
3.4.2.2 Main task loading	32
3.4.3 Virtual space	33
3.4.3.1 RT-Thread	33
3.4.3.2 Adaptation of RT-Thread	34
3.4.3.3 Virtual and physical addresses	42
3.4.3.4 Memory emulation(Data Address MMU)	43
3.4.3.5 Driver file management	43
3.5 Embedded CLI	43
3.5.1 Overview	43

3.5.2 Serial port console	43
3.5.3 Operation Guide	44
3.6 Configuration Tool	47
3.6.1 Overview	47
3.6.2 Functional framework	48
3.6.3 Installation Guide	48
3.6.4 Operation Guide	51
3.7 OTA firmware update	54
3.7.1 Overview	54
3.7.2 OTA implementation mode	55
4. Value-Added Product Development	55
4.1 Third party integrated development library document	55
5.Product application	55
5.1 Primary industry	55
5.2 Secondary industry	56
5.3 Tertiary industry	57

1. Overview

This paper mainly analyzes the composition and characteristics of the Internet of Things architecture of Brixin (Beijing) Science and Technology Co., Ltd. (hereinafter referred to as Polysense), and specifically demonstrates the plans and advantages of the Polysense architecture in the upper applications of agriculture, forestry, animal husbandry, aquaculture, fishery, industry, brewing, electrolytic aluminum industry, steel industry, smart city, smart park, cold chain logistics, health care and elderly care.

1.1 Background

1.1.1 Company introduction

Polysense Technologies was founded in Beijing in 2016. There are many offices in the world, and Luoyang factory is the global delivery place. The founding team of the company comes from well-known enterprises and research institutions in the global communication and IT industry, including Cisco, Broadcom, Intel, Hitachi, Chinese Academy of Sciences, etc. The company locates the general sensing and communication overall solution of the Internet of Things and provides intelligent terminal BYOD and OEM/ODM basic services for industrial chain partners.

The existing products and services of Brilliance are oriented to the market of enterprises, governments and operators, and have been recognized by mainstream markets and customers worldwide, and have gained effective brand recognition in China, the United States, APAC and EMEA markets.

1.1.2 Product development background

Intelligent sensor is a sensor with information processing function. The intelligent sensor has a microprocessor, which has the ability to collect, process and exchange information. It is the combination of sensor integration and microprocessor. Compared with general sensors, intelligent sensors have the following three advantages: high-precision information acquisition can be achieved through software technology, and the cost is low; Have certain programming automation ability; Diversified functions. A good "smart sensor" is a sensor and instrument package driven by a microprocessor, and has the functions of communication and on-board diagnosis.

Intelligent sensor system is a modern integrated technology, which is a rapidly developing high technology in the world today, and has not yet formed a standardized definition.

At present, the smart sensor market accounts for about a quarter of the total sensor market, and the industry is developing rapidly. Europe, the United States, Japan and other countries have a good technical foundation in the field of intelligent sensors, and the upstream and downstream supporting facilities are mature, almost monopolizing the "high, fine, and sharp" intelligent sensor market. The smart sensor industry is characterized by high technical barriers, multiple and decentralized industry segments, and the current domestic market opportunities are mainly driven by the emerging downstream applications.

With the rapid development and progress of the electronic automation industry, the sensor technology, especially the integrated intelligent sensor technology, has become increasingly active. With the rapid development of semiconductor technology, some famous foreign companies and universities are vigorously developing the research of integrated intelligent sensors. Some famous domestic universities, research

institutes and companies are also actively following up, and the integrated intelligent sensor technology has made remarkable development. Domestic smart sensors have gradually taken a step in the field of smart sensors. With the production lines and processes of military products, they have high precision, good stability and low cost. They use high-performance microcontrollers (MCU), and have both digital and analog output modes. At the same time, according to the specific needs of users (such as networking measurement, custom communication protocols), they can be re developed on the basis of the original products. The cycle is extremely short, Save time and improve efficiency for users. It has been widely used in aviation, aerospace, petroleum, chemical industry, mining, machinery, dam, geology, hydrology and other industries to measure the pressure, differential pressure, flow of various gases and fluids, and the height and weight of fluids.

However, different industrial problems will arise at different stages of the development of the Internet of Things, for example:

- **Various communication methods**

In order to meet the different requirements of different scenarios and environments, there are many communication methods of sensors, such as Zigbee, Bluetooth, Lora, NB IoT, 2G, 3G, 4G, 5G, etc. In order to access the direct connected sensors of different communication networks, the platform needs to support most of the communication methods on the market, resulting in a huge workload of platform development, increasing the complexity of platform development and the difficulty of operation and maintenance. At the same time, due to the high coupling of intelligent sensor hardware communication and sensing, it is necessary to re develop a set of products for each type of sensor communication that may be used, resulting in serious repeated product development and rigid product system that is difficult to adapt to the diverse Internet of Things market.

- **Chaotic data format**

Because there is no unified data format standard, all kinds of sensor manufacturers in the market have their own data formats, and the data analysis of different products from the same manufacturer is also very different, let alone products from different manufacturers. Therefore, it is a huge workload to manually code and analyze all sensor data formats connected to the system.

- **Heavy and redundant platform docking**

Due to the diversity and complexity of the requirements of the Internet of Things system, various sensors from different manufacturers need to be involved. At the same time, sensors from different manufacturers may be connected to their own cloud platforms. At this time, if you want to connect to the manufacturers' sensors, you need to connect to the manufacturers' cloud platforms. Data can be obtained through the cloud platform interface connection. Each manufacturer connected needs to connect to a manufacturer's data platform, The development workload is huge and the development efficiency is extremely low, resulting in long system development cycle, development difficulties and other problems.

- **Wide range of sensor models**

According to statistics, there are 26000 kinds of sensors in the world, and each sensor has its own accuracy, range, scene, etc., which makes product development need to launch many products to adapt to the above factors, corresponding to the number of sensor models in the market.

- **Serious repeated development**

The products launched in order to support different communication technologies, adapt to different application scenarios, and support different precision, measuring range, installation methods, power supply and other factors have been seriously re developed, which has also caused serious fragmentation.

1.2 Product positioning

In view of the current industrial background, IoT products, especially smart sensor products, There is an urgent need for products that can support "universal sensing and communication" to support the rapid

development of the industry. Products need a certain degree of "communication and sensor structure decoupling" to achieve, increase product versatility, solve the fragmentation of the Internet of Things, reduce product re development, reduce the difficulty of product operation and maintenance, and reduce the construction and ownership costs. The products under the iEdge4.0 architecture are products that meet the market demand!

iEdge4.0 provides the global AIoT market with one-stop proven IoT products, technologies and operation services through an open technical architecture and flexible product system. Based on the open technical architecture and product system of iEdge4.0, it provides basic product services for the market, third-party product development and flexible customization services based on 4.0, OEM/ODM services, bottom products and technical support services for third-party developers and individuals to establish their own product systems, and configurable and modular BYOD services based on 4.0 for customers and developers. An English slogan is "Build Your Own Devices (BYOD) with Polysense Technologies".

2. Product attributes

2.1 Product Matrix

In consideration of the requirements for products to be put on the market, Polysense's intelligent sensor products are named structurally and normatively according to the standards of sensor type spectrum system. Due to the wide variety of sensors, Polysense intelligent sensors adopt a logical modular structure, which divides an intelligent sensor into four parts: communication part, sensor part, main board part and power supply part. Products are classified into series according to communication technology and models according to different sensors provided, specifically:

- WxS7800 series, Based on WiFi, Smart Sensor & RTU 2-in-1 Terminal
- WxS8800 series, Based on LoRa, Smart Sensor & RTU 2-in-1 Terminal
- WxS9800 series, Based on NB-IoT (China), Smart Sensor & RTU 2-in-1 Terminal
- WxS9900 series, Based on NB-IoT (Global), Smart Sensor & RTU 2-in-1 Terminal
- WxSC800 series, Based on LTE Cat1, Smart Sensor & RTU 2-in-1 Terminal
- WxSC900 series, Based on LTE Cat1/GPS, Smart Sensor & RTU 2-in-1 Terminal
- WxSD800 series, Based on LTE Cat4, Smart Sensor & RTU 2-in-1 Terminal
- CxS1800 series, Based on Ethernet RJ45, Smart Sensor & RTU 2-in-1 Terminal

Based on the above existing product series, the product also plans to expand the series, including: WxS2800 series, mesh based smart sensors; WxS3800 series, smart sensor based on Z-Wave; WxS4800 series, intelligent sensor based on ZigBee; WxS5800 series, intelligent sensor based on Sigfox; WxSB800 series, smart sensor based on eMTC; WxSF800 series, based on 5G mMTC smart sensor; wait.

On the basis of series division, the sensors are divided into three types according to their working principles. Six digit coding is used. The first digit is used to distinguish the major categories of sensors: 1 - biomass sensor, 2 - chemical sensor, 3 - physical sensor, and 4 - mixed sensor. The last five digits are used for the working principle and product code of each type of sensor.

At present, Polysense has more than 500 sensors, mainly including the above-mentioned 2-chemical quantity and 3-physical quantity sensors. 1 - Biomass sensors are mainly medical devices and wearable devices. At present, the company has not set foot in this field. The 2-chemical quantity sensor mainly includes 1-various

gas sensors, 2-humidity sensors and 3-ion sensors. The 3-physical quantity sensor includes six parts: 1-mechanical quantity sensor, 2-thermal sensor, 3-optical quantity sensor, 4-magnetic sensor, 5-electrical sensor and 6-acoustic sensor. For example:

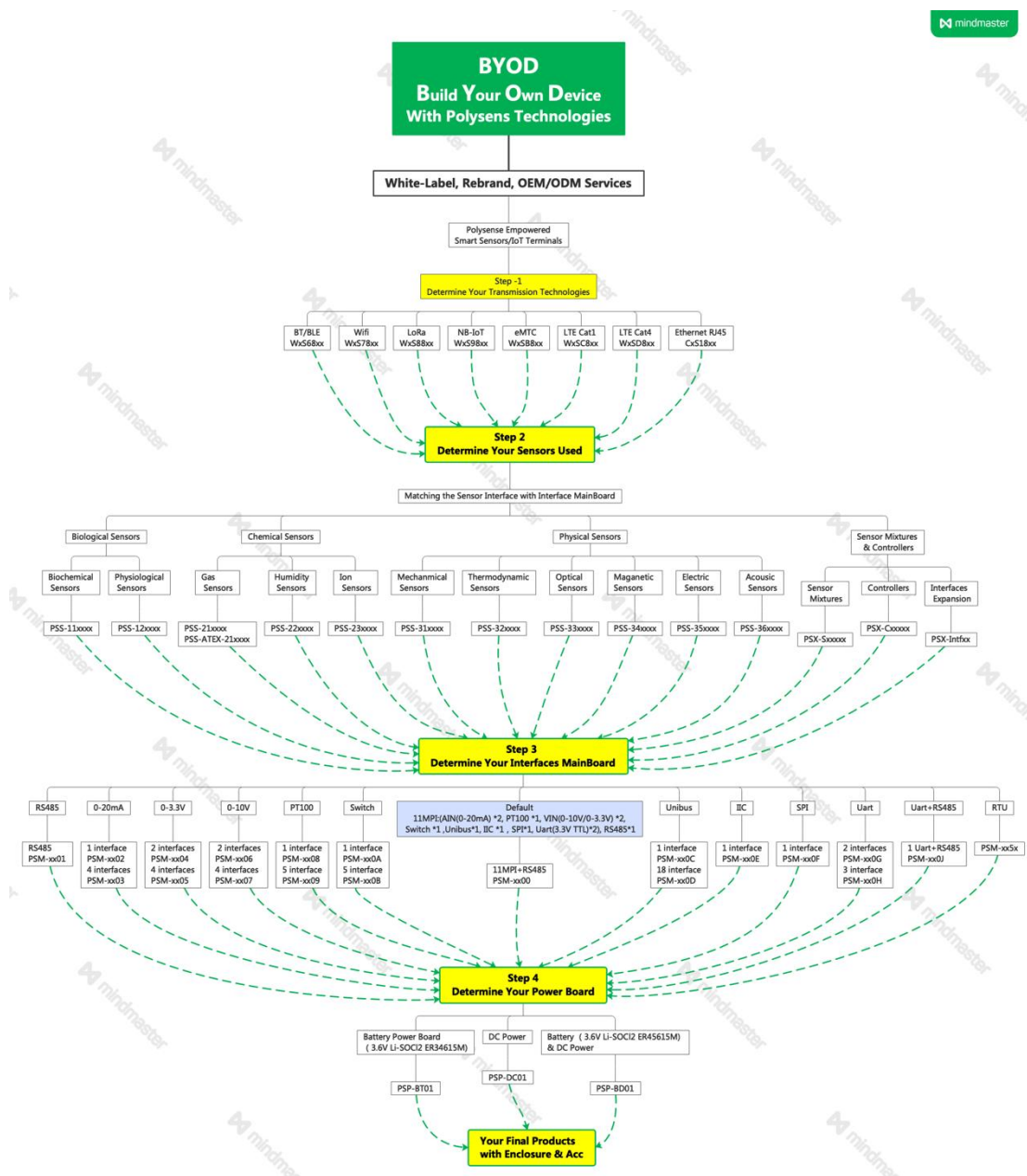
PSS-21E0J4 carbon dioxide CO2 sensor (conventional measurement range: 0-2000PPM, resolution: 1PPM, non dispersive infrared technology NDIR) LP 3.3V, PSS-362022 sound sensor (measurement range: 30-130dB, outdoor).

Based on the above rules, we can clearly know the meaning of the following product codes:

WxS8000-21E0J4 represents a LoRa based CO2 intelligent sensor; WxS9000-362022 represents an intelligent outdoor noise sensor based on NB IoT;

How to modularize the construction of an intelligent sensor? It is necessary to combine the components of a product. A typical intelligent sensor includes the following components:

- Communication technology
- Sensor
- Equipment power supply mode
- Interface to the sensor
- Equipment enclosure
- Other accessories



2.2 Product iteration

iEdge is the bottom lightweight intelligent IoT terminal OS developed by Brixin for the development of "general sensing and communication" IoT sensor products. It is similar to Alibaba Things, Huawei LiteOS, Tencent TinyOS, and open-source FreeRTOS, RT Thread. The difference of iEdge is that in order to support its general product positioning, it not only supports the basic functions of these edge intelligent terminal OS, but also introduces virtualization technology in the microkernel, so that products based on iEdge4.0 can completely shield hardware differences, including the solution of sensor interface differences through MPI interfaces and the cross CPU versatility brought by kernel virtualization, which completely solves the product requirements of modularization and configuration.

Since the launch of the first generation product in 2017, iEdge has developed three generations to iEdge3.0. Due to the new technical system and architecture adopted by iEdge4.0, the current iEdge4.0, as a new generation version, will be the cornerstone of the company's product development in the next few years.

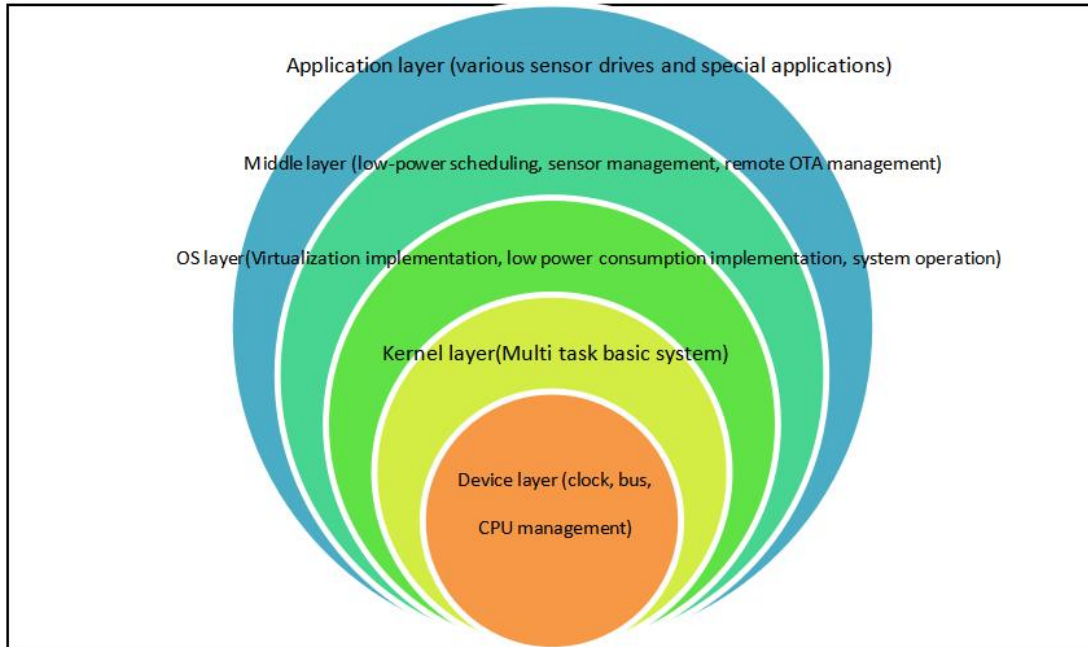
2.3 Product highlights

- The configuration and modularization of the kernel system and hardware carrier improve the flexibility of the product
- Modular design is conducive to the rapid introduction of new products, including new communication technology support and sensor support
- The kernel supports the access of third-party development equipment, and supports the business development of partners systematically
- Configured and modularized products are conducive to meeting the requirements of fragmented scenarios of the Internet of Things
- Reduce device power consumption and resource consumption
- Products with unified standards and flexible architecture can greatly reduce the operation and maintenance costs and difficulties of comprehensive projects
- Modular software kernel, which can quickly support the expansion of customer needs, such as the national security algorithm
- Configured and modularized products with dynamically loaded position machines can greatly reduce dependence on manufacturers' services
- The virtualization technology of the kernel can quickly adapt to new MCU/CPU architecture products, whether ARM or RISC-V products, or safe and independent CPU product systems
- Release computing power at the edge

3. System Architecture

3.1 System framework

Overall architecture of iEdge4.0:



3.1.1 Application layer

Based on the kernel virtualization mechanism (see the following virtualization related description for details), various drivers and logic development of the application layer support multi platform compilation and combined use. Whether it is System WorkBench, RT Thread or other various compilation modules and drivers, the compiled application layer code can be used. Users can also develop and extend special drivers or adjust some practices to meet their own needs.

3.1.2 Middle layer

The main APP is currently developed by System Workbench and can be later adapted to Keil, RT Thread or other development environments. It mainly provides the key low-power scheduling system, remote management and dynamic configuration, the position machine background and the module drive management of each sensor, which is the main logic carrier of the entire system.

3.1.3 OS layer

Not open to the public, only SDK operation API- <http://ota.polysense.online/iEdge4.0/doc/files.html>

It has the core level code simulation operation, supports the upper layer (application layer) virtual space development mechanism, realizes the decoupling of the code construction environment and the running environment, so as to facilitate the stacking of various upper layer applications. It is the core of the entire architecture. All low-power systems, underlying access management, timer triggering, message transmission between tasks, and initial system loading at the beginning of power on are completed in this layer. In addition, the core virtual atomic operations of the entire architecture are also provided here, providing the necessary support platform for the stable and flexible operation of the middle layer and application layer.

3.1.4 Kernel layer

A multi-level task scheduling and memory management system based on FreeRTOS like architecture is used to implement the most basic task environment

3.1.5 Device layer

The device bus and CPU bottom driver provide the most basic hardware access and pin definition management.

3.2 Structure and appearance

3.2.1 Reliable structural design

- The shell adopts the form of shell+supporting plate. The product with this structure is not only simple and convenient to fix, but also more convenient to maintain the product later;
- The lower shell of the shell is universal, and the upper cover adopts a variety of combinations: lower shell+sealed upper cover (protection can reach IP67), lower shell+high permeability shell, lower shell+low permeability shell, lower shell+ink screen display shell; This combination form makes the product combination more flexible and more extensive under the condition of low cost, and the whole shell has strong versatility;
- Integrate various types of sensors and batteries through optimization of internal structure space to reduce the cost of shell selection in the later stage;
- External interfaces are reserved, with up to 12 interfaces, so that the shell has a strong external connection capability, which can meet the use requirements in various situations;
- The shell material is PC+ABS+flame retardant, which makes the mechanical property, impact strength, high temperature resistance, ultraviolet resistance, flame retardancy and other properties of the whole product better, and can meet the use requirements of various occasions;
- High-capacity battery, 3.6V 14000mAH standard 34615M lithium-ion sub-battery lithium-SOCl₂; It provides high capacity and long endurance for various scenarios requiring battery power;
- Product specification parameters (as shown below):

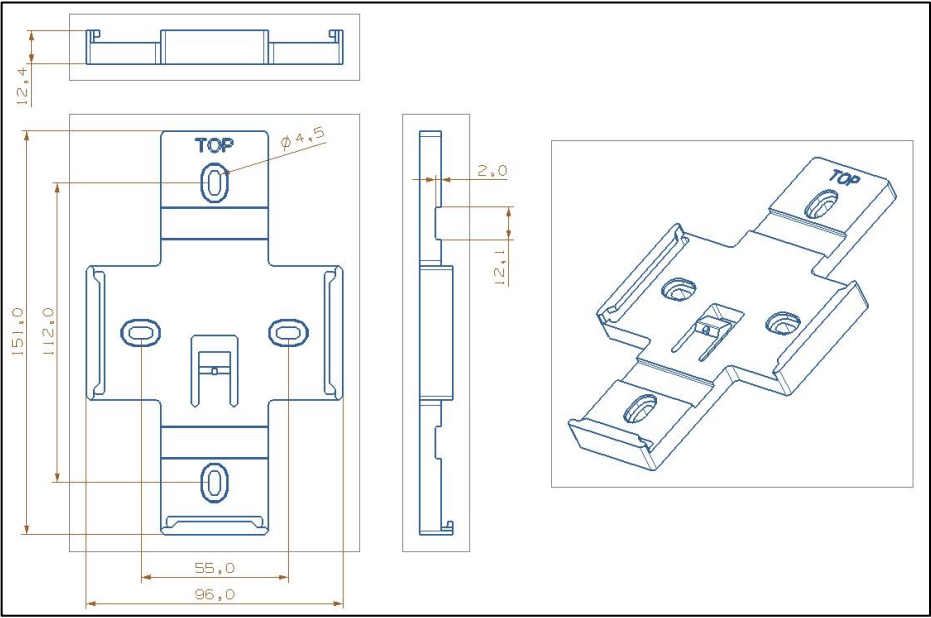


Figure 1 (Supporting plate)

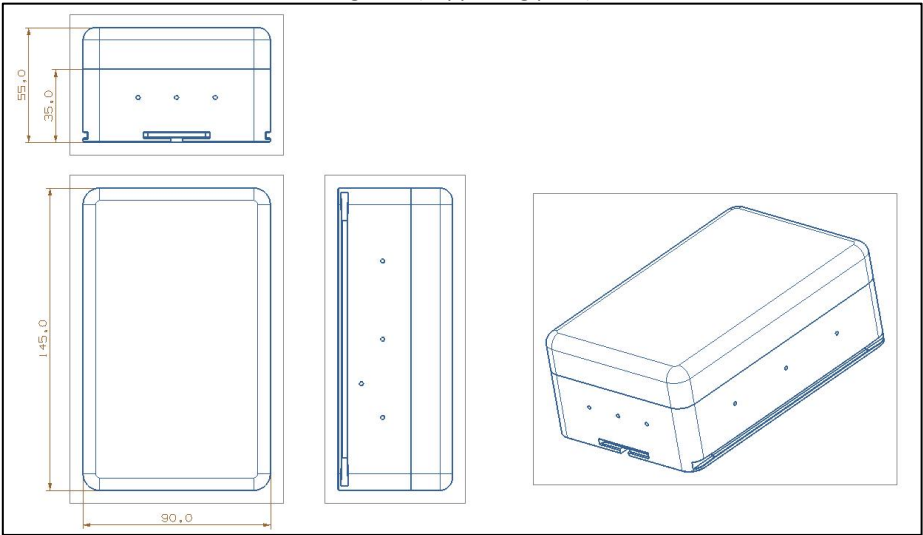


Figure 2 (sealed shell)

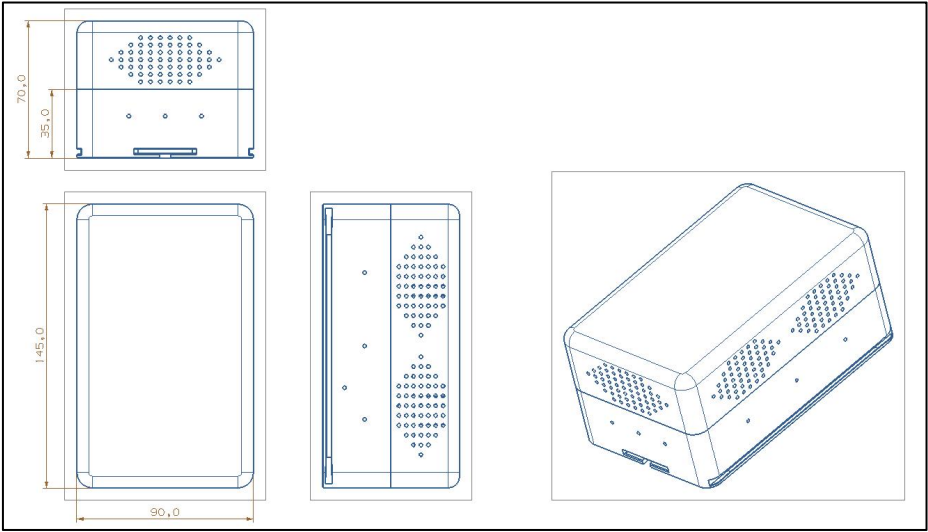


Figure 3 (High breathable shell)

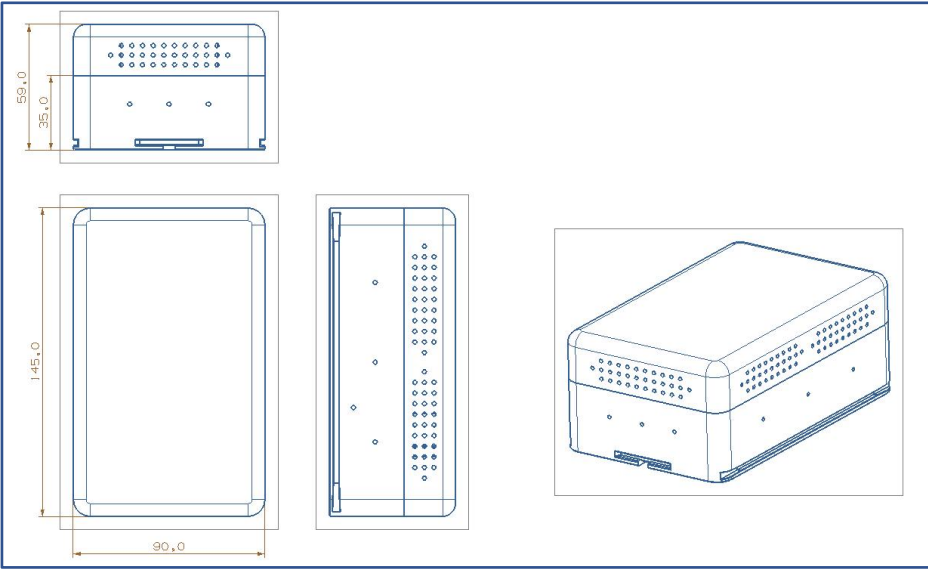


Figure 4 (Low breathable shell)

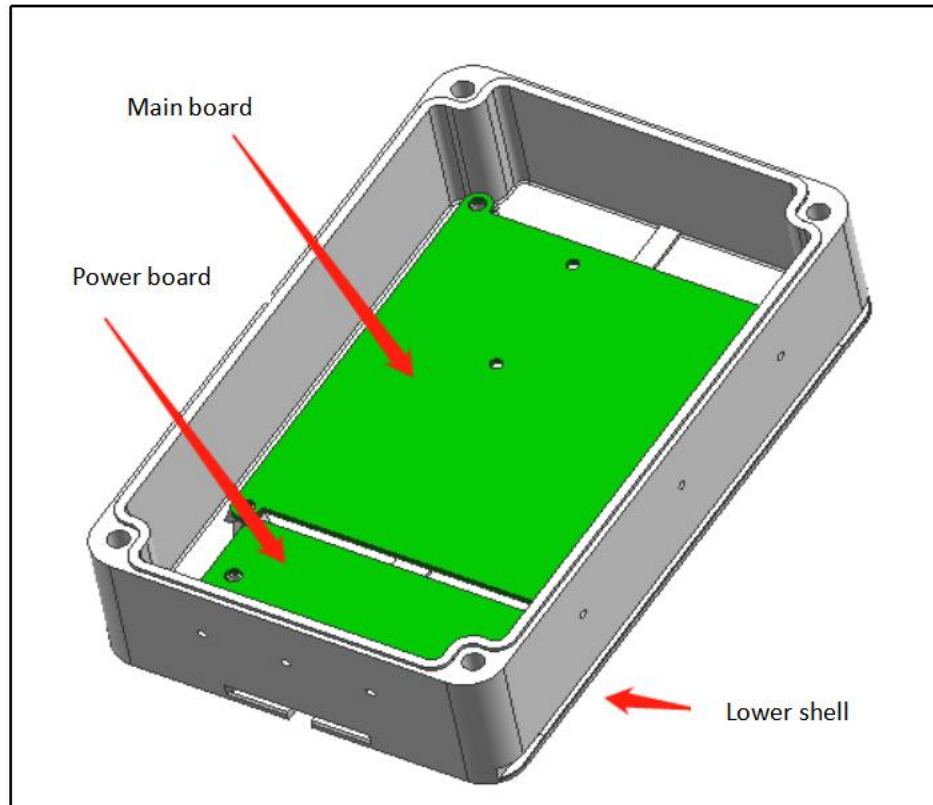


Figure 5 (Internal Structure of Lower Shell)

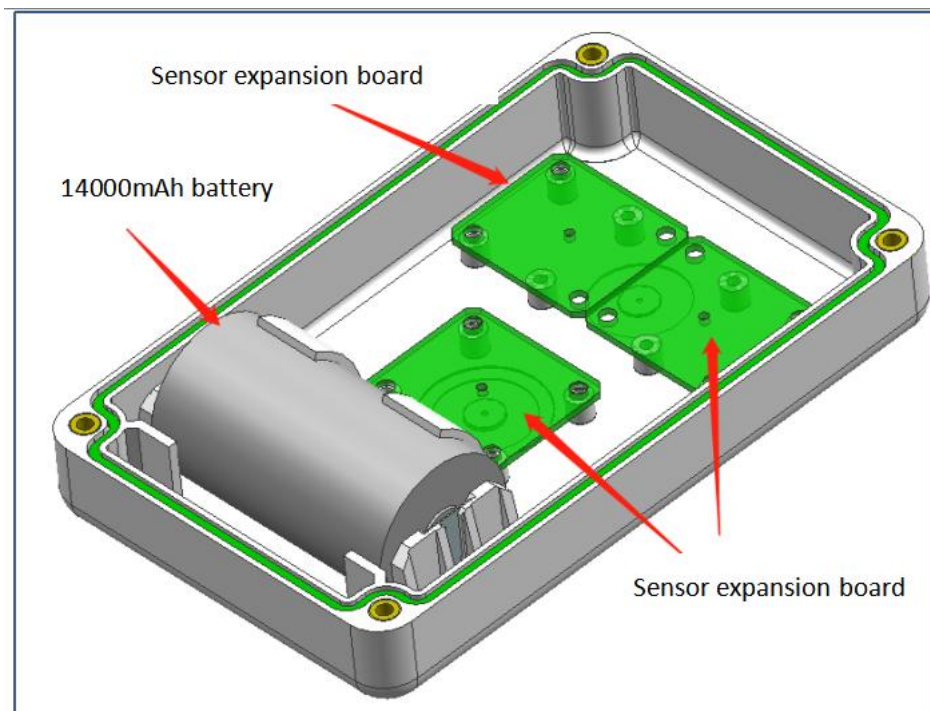


Figure 6 (Internal structure layout of upper seal cover)

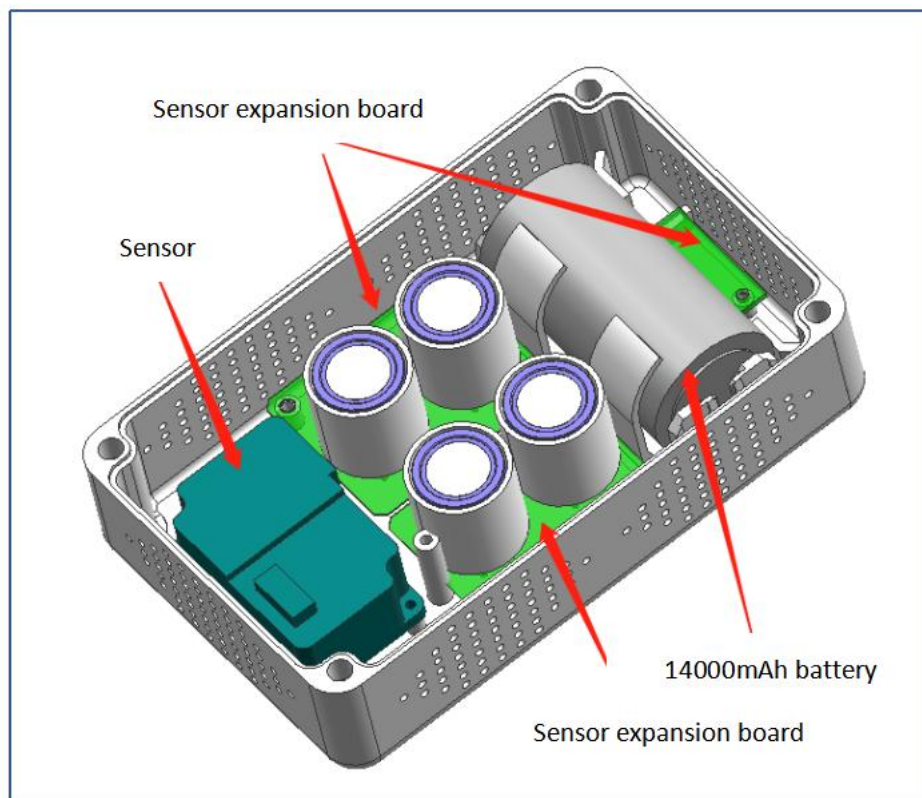
The size of sensor expansion board is square (3.2cm * 3.2cm) structure, and multiple boards can be stacked

and installed as appropriate:

- Two axis+three axis acceleration;
- GPS/Beidou positioning;
- Passive infrared PIR (indoor);
- Switching value (1/2 Call Button) (waterproof structure shall be considered when outdoor);

In addition to the above sensor expansion board, the sealed upper cover structure can also provide:

- Gas (indoor and outdoor, supporting 1/2 waterproof gas probes);
- Temperature and humidity (indoor and outdoor, with IP67 waterproof probe outside);
- Multiple external independent sensors;



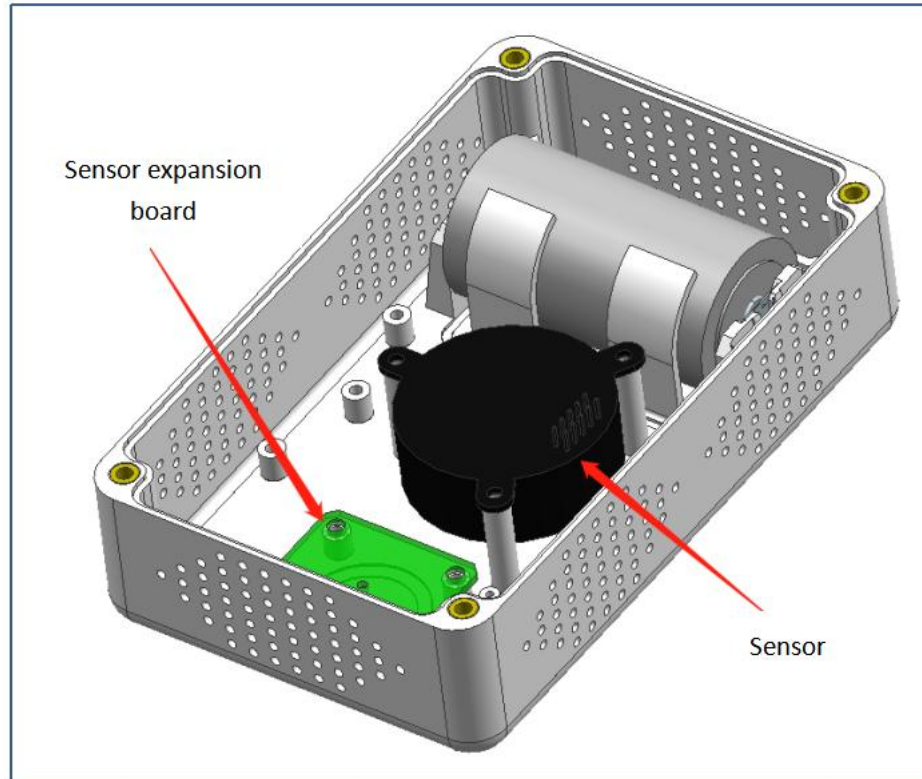


Figure 7 (Internal structure layout of high breathable upper cover)

High breathable shell structure supports AQI indoor version, including temperature and humidity, PM1/2.5/10, SO₂, NO₂, O₃, CO; In the highly breathable structure housing, in addition to supporting 4 gas probes internally, it can also provide additional gas monitoring capability through 1~2 external gas probes. Flexible structural design can be widely used to support standard AQI monitoring of indoor environment.

The internal structure of the highly breathable housing adopts two groups of gas expansion plates to support four gas probes, and each group of expansion plates can support the combination of two gases.

The highly breathable shell can also support high-precision CO₂ and another standard 3.2cm sensor expansion board, as well as temperature and humidity sensors, providing battery powered applications in indoor environments.

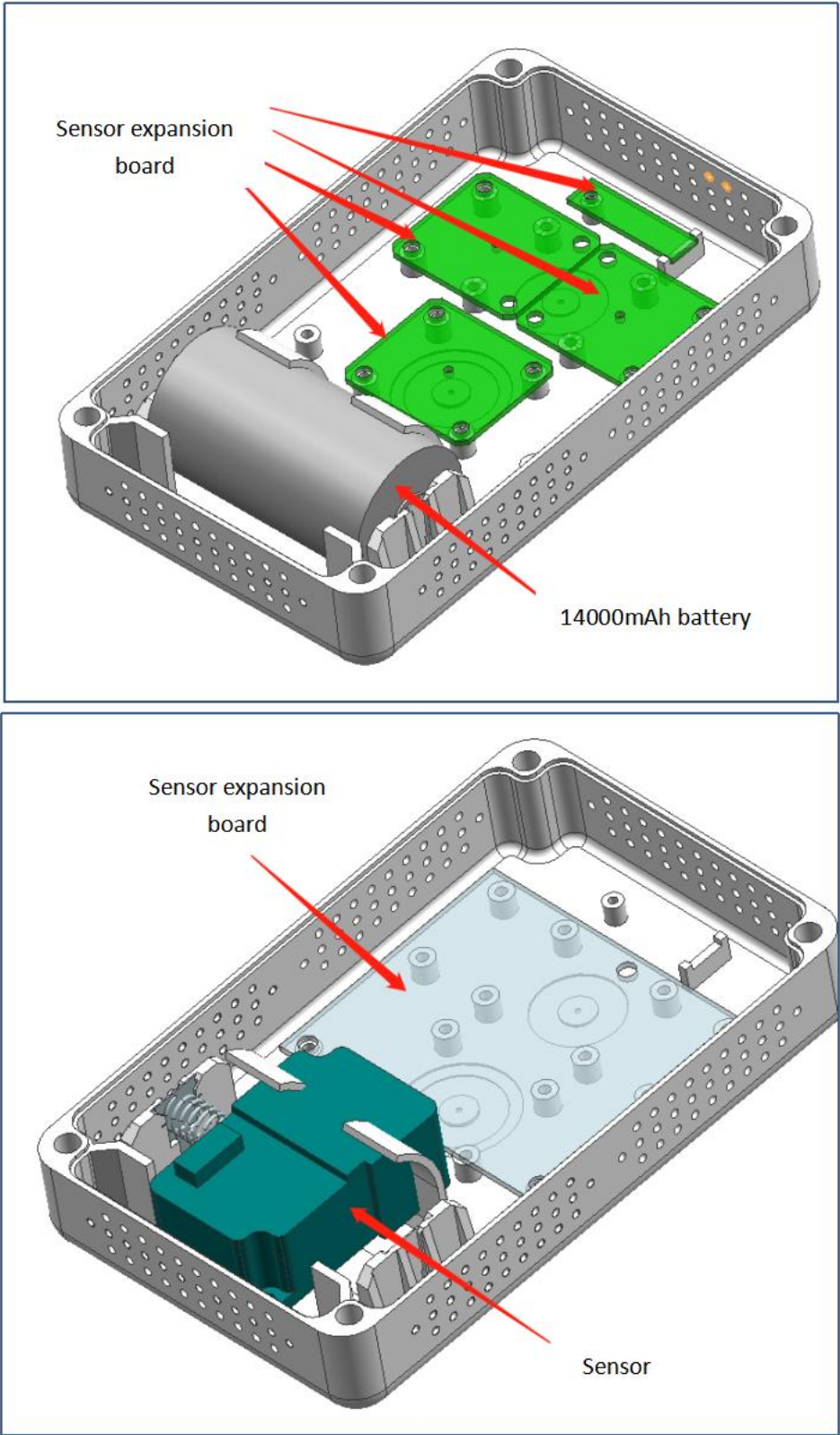


Figure 8 (Internal structure layout of low breathable shell)

The size of the sensor expansion plate is square (3.2cm * 3.2cm) structure, and the two plates can be

overlapped and installed as appropriate:

- Two axis+three axis acceleration;
- GPS/Beidou positioning;
- Passive infrared PIR;
- Switching value (1/2 Call Button)

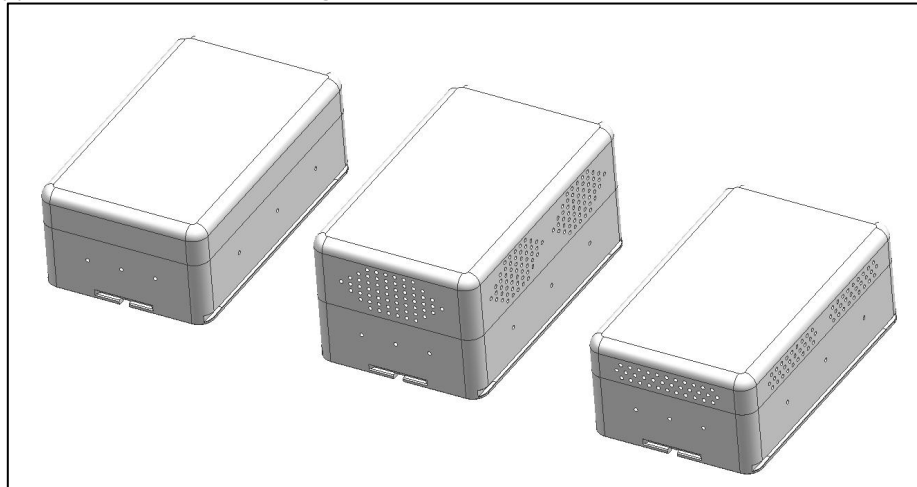
In addition to the above sensor expansion board, the low breathable upper cover structure can also provide:

- Gas (indoor and outdoor, supporting 1/2 waterproof gas probes);
- Temperature and humidity (indoor and outdoor, with IP67 waterproof probe outside);
- Multiple external independent sensors;

3.2.2 Universal design

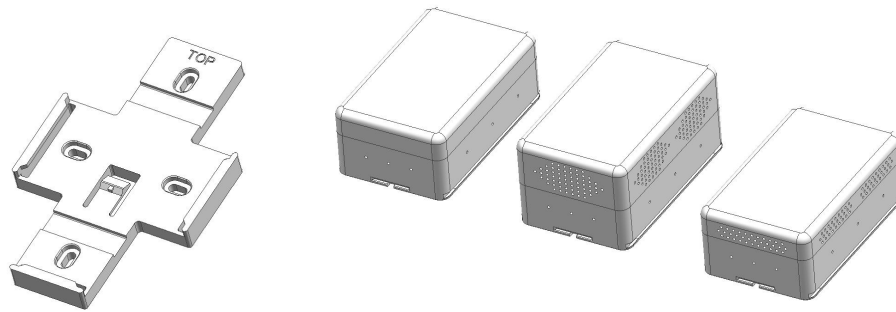
- The appearance of the product shell is designed as a cuboid shape, with the length width ratio close to the golden section, making the overall first sense more harmonious and beautiful;
- The outer surface of the shell is simple and round. Customers can carry out secondary transformation on the surface according to their own needs, which can meet a variety of customer needs;
- The air vent on the upper cover of the breathable shell is designed to be beautiful for use, and the breathable area is large, so that the internal sensor can better sense the changes of the external environment.

The product appearance is shown in the figure below:

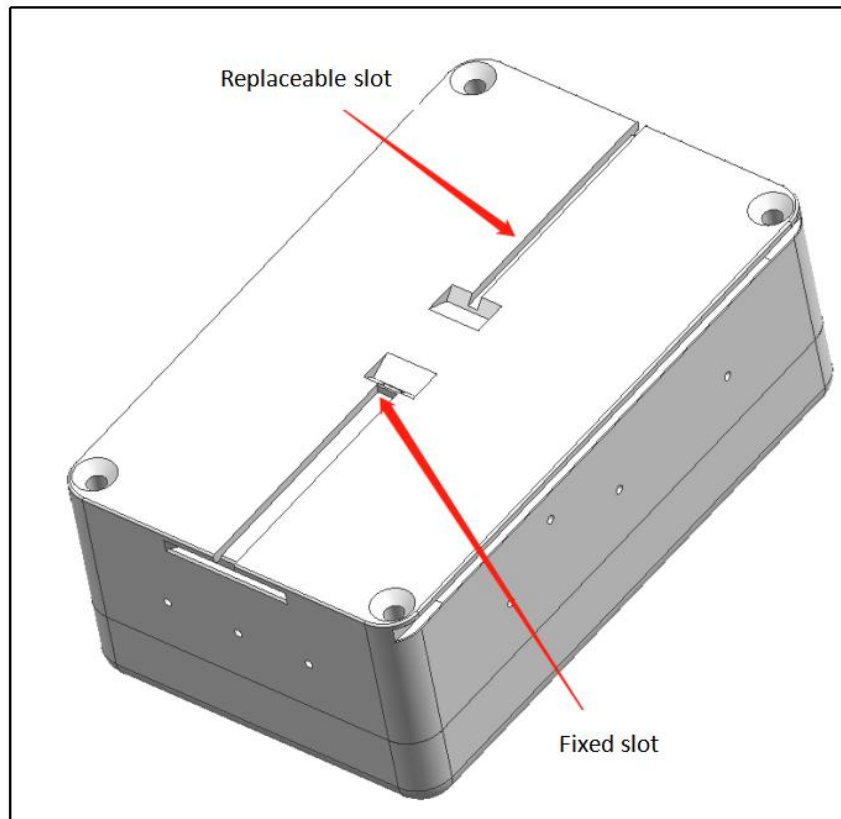


3.2.3 Installation Guide

- The product is mainly composed of two parts: shell+pallet, as shown in the figure below:



- The installation of the support plate is divided into screw fixing (4 M4 screw fixing holes) and hoop mounting (two 12mm hoop slots). The customer selects the installation method according to his own situation to fix the support plate. Since the shell needs to be inserted into the support plate, the flat surface should be selected for fixing the support plate to prevent the shell from being inserted due to serious deformation of the support plate;
- There are card slots at the bottom of the shell, which are divided into reusable slots and fixed slots. Customers can choose the insertion direction according to their own use scenarios to meet their own use needs, as shown in the following figure:



○

3.3 Hardware

3.3.1 Technical specifications

3.3.1.1 MCU

The new generation of products adopts domestic MCU, which is produced by Huada Semiconductor Co., Ltd., a professional integrated circuit development platform company under China Electronics Industry Group Co., Ltd. (CEC): HC32L196KCTA.

- CPU core: ARM Cortex-M0 CPU
- Maximum dominant frequency: 48MHz
- Operating voltage range: 1.8V~5.5V
- Program FLASH capacity: 256KB
- RAM total capacity: 32KB
- Peripherals/functions/protocol stack: on-chip temperature sensor; DMA; watchdog; Low voltage detection; Hardware cryptographic algorithm engine; CRC calibration; PWM; LCD/LED drive; True random number generator; RTC real-time clock
- Operating temperature range: - 40 °C~+85 °C

3.3.1.2 Universal main board

The overall dimensions are shown in the figure below:

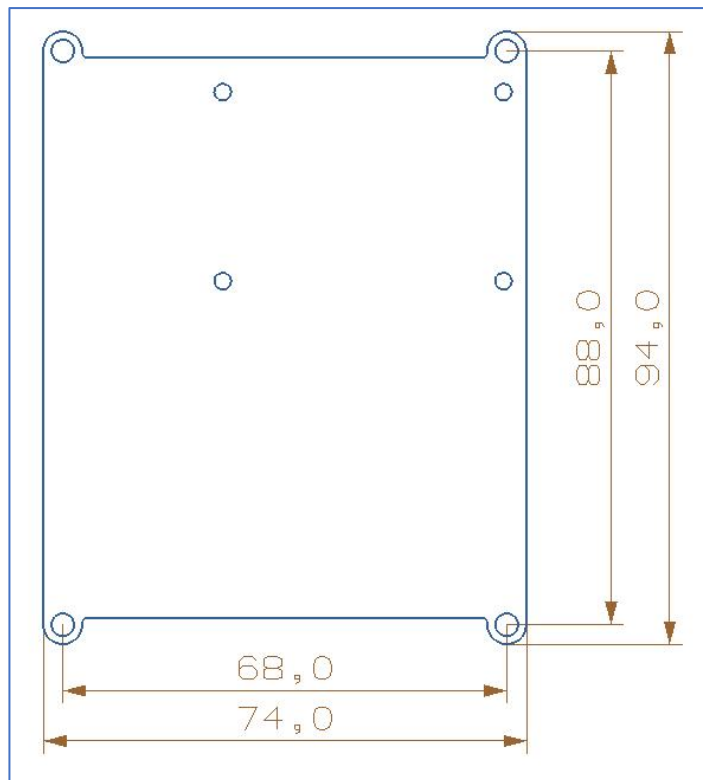


Diagram of the LoRa module showing various interfaces and components:

- USB to serial port
- Analog quantity 0~10V
- Analog quantity 4~20mA
- Download debugging interface
- PT100 temperature interface
- Digital quantity input
- Two way OD output
- Battery interface
- User key
- Communication board: can connect LoRa/NB-IOT/WIFI/BLE/Cat1.8c
- Green indicator 1
- Red indicator 2
- Power board interface
- 18B20 interface
- 5V output
- 3.3V output
- Uart2 interface
- RS485 interface

- There are 19 different interface combination masters for the iEdge4.0 series products. By default, full function combination interfaces are provided, as shown below:

PSM-xx00	11MPI	RS485+OD&PWR	WxS Terminal Main Board MPI Interfaces-Any Mixture of Following: MPI:(AIN(0-20mA) *2, PT100 *1, VIN(0-10V/0-3.3V) *2, Switch *1,Unibus*1, IIC *1, SPI*1, Uart(3.3V TTL)*2), RS485*1: OD*2, PWR*2
PSM-xx01	-	RS485+OD&PWR	WxS Terminal Main Board RS485*1: OD*2, PWR*2 MoQ=500
PSM-xx02	MPI (0-20mA)*1	OD&PWR	WxS Terminal Main Board MPI (0-20mA)*1: OD*2, PWR*2 MoQ=500
PSM-xx03	MPI (0-20mA)*4	OD&PWR	WxS Terminal Main Board MPI (0-20mA)*4: OD*2, PWR*2 MoQ=500
PSM-xx04	MPI (0-3.3V)*2	OD&PWR	WxS Terminal Main Board MPI (0-3.3V)*2: OD*2, PWR*2 MoQ=500
PSM-xx05	MPI (0-3.3V)*4	OD&PWR	WxS Terminal Main Board MPI (0-3.3V)*4: OD*2, PWR*2 MoQ=500
PSM-xx06	MPI (0-10V)*2	OD&PWR	WxS Terminal Main Board MPI (0-10V)*2: OD*2, PWR*2 MoQ=500
PSM-xx07	MPI (0-10V)*4	OD&PWR	WxS Terminal Main Board MPI (0-10V)*4: OD*2, PWR*2 MoQ=500
PSM-xx08	MPI (PT100)*1	OD&PWR	WxS Terminal Main Board MPI (PT100)*1: OD*2, PWR*2 MoQ=500
PSM-xx09	MPI (PT100)*5	OD&PWR	WxS Terminal Main Board MPI (PT100)*5: OD*2, PWR*2 MoQ=500
PSM-xx0A	MPI (Switch Output)*1	OD&PWR	WxS Terminal Main Board MPI (Switch Output)*1: OD*2, PWR*2 MoQ=500
PSM-xx0B	MPI (Switch Output)*5	OD&PWR	WxS Terminal Main Board MPI (Switch Output)*5: OD*2, PWR*2 MoQ=500
PSM-xx0C	MPI (Unibus)*1	OD&PWR	WxS Terminal Main Board MPI (Unibus)*1: OD*2, PWR*2 MoQ=500
PSM-xx0D	MPI (Unibus)*18	OD&PWR	WxS Terminal Main Board MPI (Unibus)*18: OD*2, PWR*2 MoQ=500
PSM-xx0E	MPI (IIC)*1	OD&PWR	WxS Terminal Main Board MPI (IIC)*1: OD*2, PWR*2 MoQ=500
PSM-xx0F	MPI (SPI)*1	OD&PWR	WxS Terminal Main Board MPI (SPI) *1: OD*2, PWR*2 MoQ=500
PSM-xx0G	MPI (Uart)*2	OD&PWR	WxS Terminal Main Board MPI (Uart TTL) *2: OD*2, PWR*2 MoQ=500
PSM-xx0H	MPI (Uart)*3	OD&PWR	WxS Terminal Main Board MPI (Uart TTL) *3: OD*2, PWR*2 MoQ=500
PSM-xx0J	MPI (Uart)*1	RS485+OD&PWR	WxS Terminal Main Board MPI (Uart TTL) *1, RS485*1: OD*2, PWR*2 MoQ=500

The default full function master provides comprehensive sensor interface capabilities and can adapt to most scenarios. The interfaces provided are as follows:

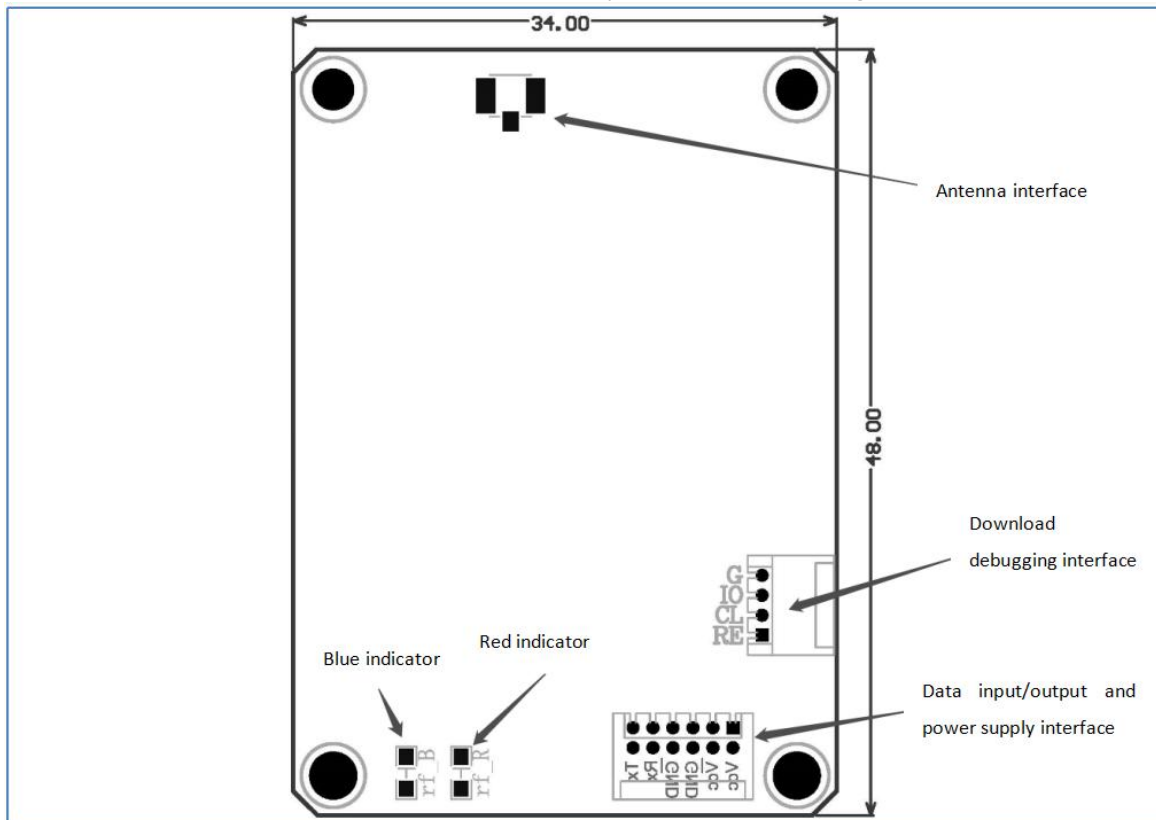
- AIN (- 0-20mA) 2 channels, 4 channels at most;
- PT100: 1 channel by default, 5 channels at most;
- VIN (0-10V/0-3.3V): 2 channels by default, 4 channels at most;
- Switching value: 1 channel by default, 5 channels at most;
- Single bus: 1 channel by default, 18 channels at most;
- I2C: 1 channel by default, 1 channel at most;
- SPI: 1 channel by default, 1 channel at most;
- Serial port ttl: 2 channels by default, 3 uarts at most;
- RS485: 1 channel by default, 1 channel at most;

The main board provide a variety of different interfaces required by sensors, and is the core operating carrier of the entire 4.0 technology architecture, iEdge OS.

3.3.1.3 Communication board

3.3.1.3.1 LoRa communication board

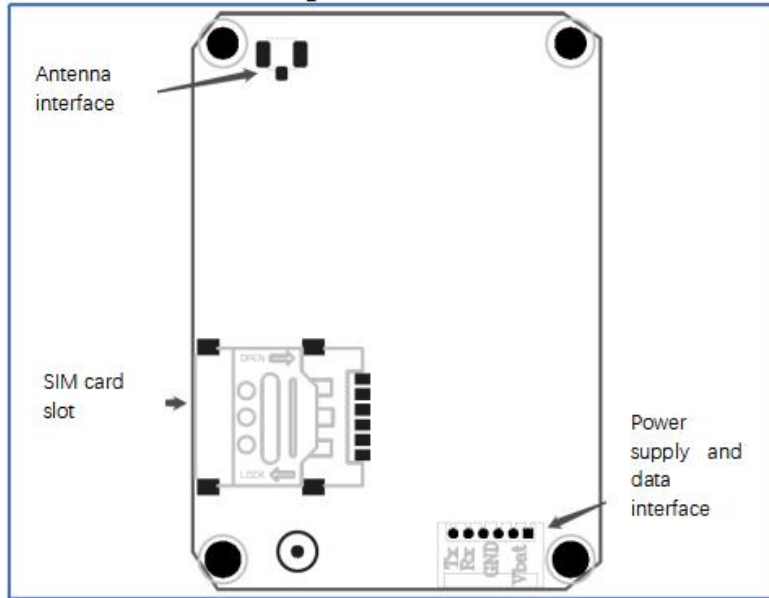
The overall dimension is 34 * 48mm, and the interface layout is shown in the figure below:



- The main control chip of oLoRa communication sub board adopts domestic Huada ultra-low power consumption MCU
- LoRa chip adopts Semtech SX1276 chip
- Full power transmission+20dBm
- 3.3V voltage supply
- Maximum instantaneous current: about 270mA
- Ultra low sleep current<5 μ A
- Support LoRaWAN protocol
- Operating ambient temperature: - 20~85 $^{\circ}$ C

3.3.1.3.2 NB communication board

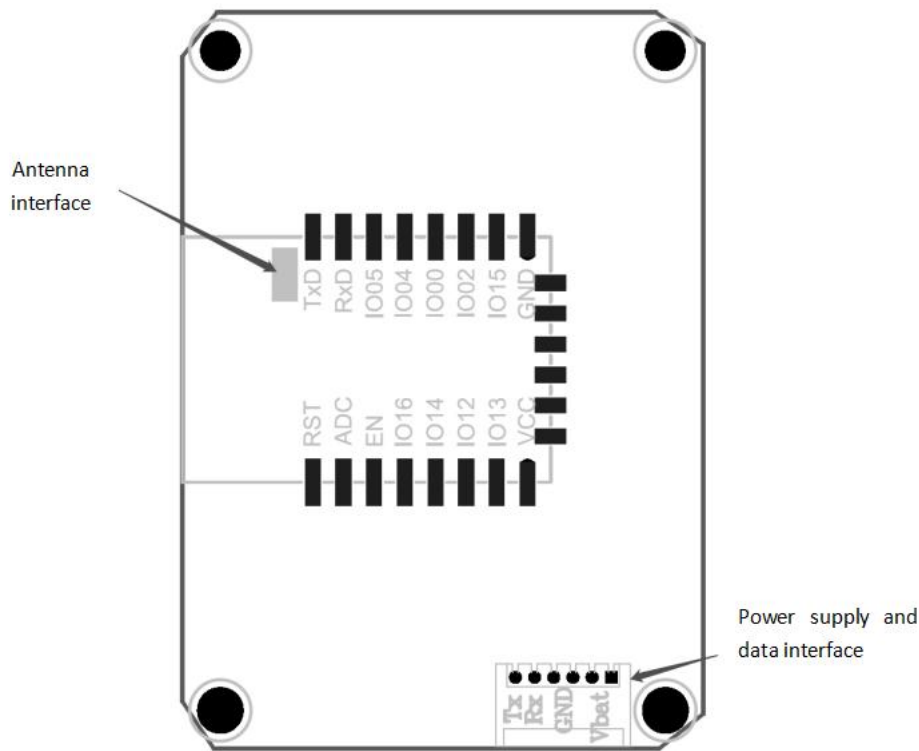
The overall dimension is 34 * 48mm. The interface layout is shown in the figure below:



- Support various NB communication modules:
 - Remote BC28
 - Remote BC68
 - Move BG95
 - Youfang N306
- The NB communication board is powered by 3.3V voltage
- Transmission instantaneous peak current 280mA (different according to network conditions)
- The oSIM card slot supports the nano card, and supports the networks of mobile, China Unicom and China Telecom
- BC68 and BG95 support the network of foreign operators
- Operating ambient temperature: - 20~85 °C

3.3.1.3.3 WIFI communication board

The overall dimension is 34 * 48mm. The interface layout is shown in the figure below:



- Antenna can be onboard microstrip antenna or IPEX external antenna.
- Operating voltage: 3.3V,
- Sleep current 20 μ A,
- Firmware supports cloud OTA upgrade,
- Support STA/AP/STA+AP working mode
- Operating ambient temperature: - 20~85 $^{\circ}$ C

3.3.1.3.4 LTE Cat1 communication board

The overall dimension is 34 * 48mm. The main parameters are as follows:

- The combined Air724UG communication module is adopted,
- External 4G antenna,
- Support WiFi Scan, WiFi hotspot scanning and WiFi positioning,
- Classic Bluetooth/BLE support,
- Supported frequency band:
 - LTE-FDD B1/B3/B5/B8
 - LTE-TDD B34/B38/B39/B40/B41
- Transmission power
 - LTE-FDD: Class3(23dBm+-2dB)
 - LTE-TDD: Class3(23dBm+1/-3dB)
- LTE characteristic

- LTE-FDD:Maximum uplink rate 5Mbps, maximum downlink rate 10Mbps
- LTE-TDD:Uplink and downlink configuration 1: maximum uplink rate 4Mbps, maximum downlink rate 6Mbps
- LTE-TDD:The uplink and downlink configuration 2 has a maximum uplink rate of 2Mbps and a maximum downlink rate of 8Mbps
- Support network protocols:
TCP/UDP/PPP/FTP/HTTP/NITZ/CMUX/NDIS/NTP/HTTPS/PING/FTPS/FILE/MQTT
- Support mobile chip SIM card
- Normal operating temperature: - 35 ° C~+70 ° C

3.3.1.3.6 RS232 board

- Operating voltage 3.3V
- MAX3232 chip and charge pump principle are used to convert 3.3V uart serial port to standard $\pm 12V$ three wire RS232 interface

3.3.1.4 Batterypower board

- Used with 4.0 main board,
- Function 1. DC power supply voltage reduction;Input voltage 5~24V;Output voltage, 5V/1A, 3.3V/2A,
- Function 2. Connect various suitable batteries to power the equipment

3.3.1.5 Antenna

Different antennas can be selected according to different sub boards. The following types of antennas can be selected:

- Spring antenna (LoRa, NB only)
- FPC patch soft antenna (applicable to various sub boards, which can be pasted on the internal shell of equipment, etc.)
- External glue stick antenna (applicable to various sub boards)
- External sucker antenna with extension cable (applicable to various sub boards).

3.3.1.6 Hardware interface

❖ J2/J3 interface

Port	Definition	General functions	Default function
J3 interface	Ai1	4-20mA, 0-10V, 0-3.3V, Digital input/output	4-20mA
	Ai2	4-20mA, 0-10V, 0-3.3V, Digital input/output	4-20mA
	Gnd	Ground	GND
	Ai2	4-20mA, 0-10V, 0-3.3V, Digital input/output	0-10V
	Ai2	4-20mA, 0-10V, 0-3.3V, Digital input/output	0-10V
	GND	Ground	GND
J2	BusV	18B20Power supply (controllable)	

interface	Data	18B20data line	
	Gnd	Ground	

❖ P1 interface

Communication board interface.

❖ J5 interface

Port	Definition	General functions	Default function
J6 interface (RS485)	A	485 data line	
	B		

❖ J9 interface

Port	Definition	Function
J9 interface	RTS2	Uart2-RTS(It can be used as an IO port once)
	CTS2	Uart2-CTS(It can be used as an IO port once)
	Tx2	UART2 Serial port output
	Rx2	UART2 Serial port input
	GND	GND

❖ J8/J9 interface

Port	Definition	General functions	Default function
J8 interface	OD3	As an output port (open drain, can be pulled up to 3.3V)	
	OD5	As an output port (open drain, can be pulled up to 5V)	
	Gnd	Ground	GND
J9 interface	PWR3	3.3V controllable output	
	GND	Ground	
	PWR5	5V controllable output	
	GND	Ground	

❖ J7/RPT100 interface

Port	Definition	General functions	Default function
J7 interface	Di1	Switch input	
	Gnd	Ground	GND
RPT100 interface	+	PT100 input positive	
	+	PT100 input positive	

	-	PT100 input negative	
--	---	----------------------	--

- ❖ USB port, connected to type-C male connector, used for program updating and debugging

3.3.2 Scalability

The following interfaces can be extended on the basis of the general main board:

- Single RS485
- Single 1-way default AIN (0-20mA)
- Single 4-way AIN (0-20mA)
- Single 1-way default PT100
- Single 5-way PT100
- Single switching value 1 channel default
- Single switching value 5 channels
- RS485+Uart TTL
- Single UART TTL
- Single I2C
- Single bus default 1 way
- Single bus 18 channels
- Single SPI

3.3.3 Certification report

Existing product certification includes CE, RoHS, FCC/IC. The certification of related products will continue to be promoted in the future.

3.4 Software

The iEdge 4.0 software provides the following main functions:

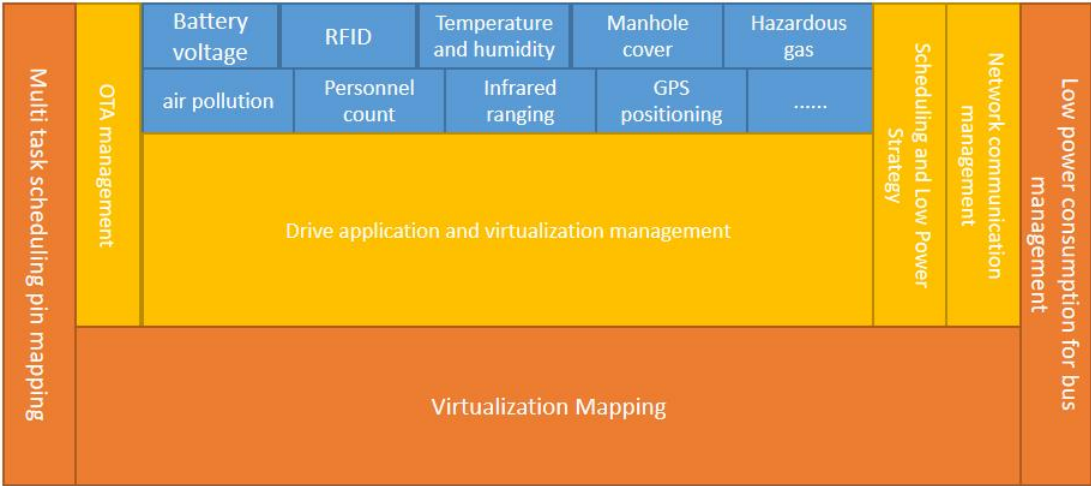
- Low power consumption system
- OTA management
- Sensor management for cycle strategy
- Host computer background
- Built in CLI
- Virtualization

The software core is a set of virtualized multi task support platform. With the necessary support provided by the OS layer and the kernel layer, the middle layer constructs an application management platform based on virtual address space to support various driver codes and special applications generated under various or even different compilation environments. With this feature, it can be realized on a very small platform:

- Third party function modification after deployment
- Flexible driver support management
- Remote OTA upgrade and online host computer upgrade for single drive or application

3.4.1 System architecture

3.4.1.1Architecture diagram



3.4.1.2 System characteristics

In order to realize the design concept of universal sensing and communication, the following features are realized from the product software architecture:

1. Self developed iEdge kernel of IoT terminal operating system;
 - Micro kernel design
 - Multithreaded scheduling
 - Multi task priority and preemptive triggering
 - Unified Memory Manager
 - Support user-defined interrupt and multi-level preemptive interrupt priority
 - Support user-defined tasks and sensor driven development
2. Any application interface technology MPI (Multiple Purpose Interfaces) is adopted to support multiple interfaces connecting sensors, including AIN (0-20mA), PT100, VIN (0-10V/0-3.3V), switching value, single bus, I2C, SPI, serial port and RS485. Support sensors with various interfaces without changing hardware or with few changes to increase the universality of interfaces;
 - Up to 16 GPIO digital input controls
 - Up to 16 GPIO digital output controls
 - Up to 16 channels of MPI control (analog sampling or single wire drive)
 - Up to 16 I2C drives
 - Up to 16 SPI drives
 - Up to 16 serial ports (RS232 or RS485)
3. The kernel adopts platform abstraction technology to shield the differences of different CPU pins and increase the universality across CPUs; The product supports ARM, RISC-V and other CPUs of different architectures;
4. Local management capability of the system:
 - Channel local configuration serial port (default 115200 baud rate)
 - Support basic command operation of Configuration Tool and local serial port
 - Support various local firmware upgrades

- Support various parameter configuration modifications
- Support configuration export
- Support firmware export
- 5. Remote management capability of the system:
 - Support LoRa, NB-IoT, 4G/LTE, WIFI, BT/BLE and other communication protocols
 - Support expandability, including user-defined communication protocol capability
 - Support remote firmware upgrade and configuration modification
 - Support MQTT data reporting and management
 - Support JSON text data reporting

3.4.2 iEdge OS kernel

3.4.2.1 Boot OS

- Basic IO initialization and CPU frequency management
- Complete the initialization of the local configuration serial port and detect whether there is a configuration request from the Configuration Tool
- Provide basic drivers for all interfaces
- Provides all heap memory management
- Provide all interrupt hardware encapsulation and underlying calls to ensure smooth connection between user interrupt and physical interrupt
- Support local configuration of all modules (including main task)
- Support all firmware local version updates
- Support Boot OS self erasing and updating
- Support basic low power management
- Transition of management authority through main tasks

3.4.2.2 Main task loading

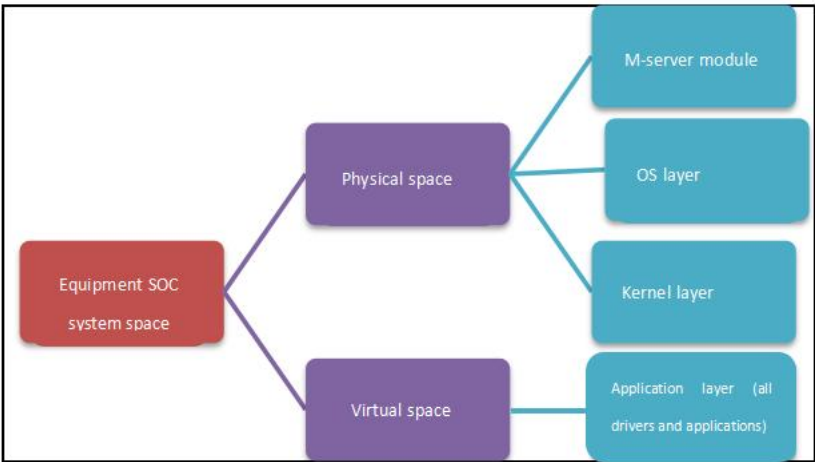
- Support at least eight sensors at the same time
- Implement periodic acquisition, reporting, and low-power power management strategies
- Support user-defined sensor driver
- Provide unified data reporting and management for all sensor drivers, including user-defined area drivers
- Provide unified JSON data uplink encapsulation
- Provide downlink JSON data parsing
- Support various communication protocol modules (including but not limited to LoRa, NB-IoT, LTE, WIFI)
- Support user-defined communication protocol module
- Support embedded MQTT client
- Support embedded MQTT reporting management or third-party MQTT reporting management (included with the module)
- All parameters support remote configuration
- Support remote OTA to upgrade firmware

3.4.3 Virtual space

For low power consumption and micro systems with limited space but various applications, there are generally only 128KB ROM and 8KB or 16KB RAM. However, in the field of the Internet of Things, there are hundreds or thousands of sensor drivers. In addition, there are various communication carriers (LoRa, WIFI, LTE, NBIOT, BLE). Almost every project is cut and customized. It is difficult to conduct on-site modification and secondary development on the deployed platform. In order to solve this contradiction, a micro system can dynamically select and even adjust applications after deployment. At this level, we first introduced the concept of virtual space to solve this problem through virtualization.

The essence of virtual space is to provide a non-existent address space on the memory RAM and permanent memory ROM provided by the physical MCU. When an application (driver) compiles and finally links in this space, the final object code it produces can be flexibly loaded and managed without relying on a specific running address pointer.

In order to support this, the OS bottom layer needs to provide a mapping from virtual address to actual physical address, and maintain this mapping relationship at runtime so that codes stored in different physical addresses can run normally in their own virtual space. At present, the modules using physical and virtual spaces are distributed as follows:



3.4.3.1 RT-Thread

RT Thread is a technology platform that integrates the real-time operating system (RTOS) kernel, middleware components and developer community. It was developed under the leadership of Mr. Xiong Paxiang and with the strength of the open source community. RT Thread is also an Internet of Things operating system with complete and rich components, high scalability, simple development, ultra-low power consumption and high security. RT Thread has all the key components required for an IoT OS platform, such as GUI, network protocol stack, secure transmission, low-power components, and so on. After 15 years of cumulative development, RT Thread has already had the largest embedded open source community in China, and is widely used in energy, vehicle, medical, consumer electronics and other industries, with a cumulative installed capacity of more than 1.4 billion. It has become the most mature and stable open source RTOS independently developed by Chinese people and the largest installed capacity in China.

RT Thread has a good software ecology. It supports all mainstream compiling tools on the market, such as

GCC, Keil, IAR, etc. The tool chain is complete and friendly. It supports various standard interfaces, such as POSIX, CMSIS, C++ application environment, Javascript execution environment, etc., which facilitates developers to transplant various applications. Commercially, it supports all mainstream MCU architectures, such as ARM Cortex-M/R/A, MIPS, X86, Xtensa, C-Sky, RISC-V, and almost all mainstream MCU and Wi-Fi chips in the market.

RT Thread Studio is recommended for RT Thread development, mainly including project creation and management, code editing, SDK management, RT Thread configuration, build configuration, debug configuration, program download and debugging and other functions. Combined with graphical configuration system, software package and component resources, it can reduce repetitive work and improve development efficiency. The link to download the latest version on the official website is as follows:

<https://www.rt-thread.org/page/studio.html>

3.4.3.2 Adaptation of RT-Thread

RT Thread Studio is the core tool for RT Thread development. Like System Workbench, it is developed based on the eclipse platform. Its interface design and style are inherited from eclipse, with less repetitive work and high development efficiency. For most MCU architectures, there is a good compilation support environment and module library.

When using RT Thread Studio to develop iEdge 4.0 application layer drivers or special logic, you need to pay attention to the following. Similarly, other systems such as System Workbench can also be referred to:

- Ensure that the compilation ROM space starts from 0x80,000,000
- Ensure that the compilation RAM space starts from 0xC0000000
- By global variable `g_export` performs function declaration

This declaration can provide virtual function entry to the bottom layer, and the bottom layer can map to the actual space of the runtime to realize the final effective operation of the program. The following is a declaration example of battery voltage acquisition drive:

```
#define DRV_VBAT_VERSION 0x0106

#define DRV_VBAT_NAME "VBAT"

ma_drv_export_t g_export = {

    .version = DRV_SENSOR_VERSION(MA_SENSOR_VBAT, DRV_VBAT_VERSION),

    .name = DRV_VBAT_NAME,

    .u.sensor={

        .power_set = sensor_vbat_set,

        .collect = sensor_vbat_collect,

    },

}
```

```
};
```

- Divide g_ You must not use additional global variables other than export
- Additional local static variables (including global variables and local variables) shall not be used
- The bottom function callback (including interrupt callback function) needs to use the translated function mapping pointer (i.e. physical pointer) provided by the bottom layer, and the function calls in the application layer can directly call virtual functions without this convention.

On micro systems, the memory is tight, especially when multiple virtual applications are activated. We know that static variables and global variables are pre allocated at compile time. If there are many such variables during the running of such multiple programs, they will inevitably occupy a large amount of memory space. Therefore, it is recommended to use the system heap memory to dynamically allocate and release memory to maximize memory utilization. The following is an example of a personnel count sensor. This example uses heap memory allocation, and releases the allocated memory when the module is unloaded. The callback function registered to the interrupt uses the mapped address provided by the bottom layer:

```
pos_status_t sensor_fastcnt_init(pos_u32_t load) {

    drv_api_t *drv = g_drv;

    irq_param_t *param;

    ma_sensor_ctrl_t *s = g_drv->s;

    pos_gpio_pin_t pin;

    pin = drv->board->pin_map(MB_PT_DIO, s->slot->io);

    if( drv->cfg->ctrl & MA_CFG_CTRL_DRV_LOG ) {

        drv->log->data("fastcnt init", load);

    }

    if( load ) {

        s->data.u32 = 0; /* clear counter by default */

        param = drv->os->malloc(sizeof(*param));

        if( !param )

            return POS_STATUS_E_MEM;

        param->pin = pin;

        drv->os->memcpy(&param->drv, drv, sizeof(*drv)); /* clone a g_drv */

        s->drv_buf = param; /* record to drv_buf */
    }
}
```

```

/*

* turn on irq

* warning: as GOT is implemented in DATA and DATA is not supported in mmap()

* all GOT based pointer can NOT be used. So the IRQ SET should use the mapped

* function pointer here (the export structure).

* the original sensor_fastcnt_irq() should NOT be used directly.

*/

drv->os->gpio->irq_set(

    pin,

    POS_GPIO_MODE_INPUT_PU+POS_GPIO_MODE_IRQ_FALLING,

    (pos_func_t)drv->s->drv.u.sensor.irq_cb, /* has to be this. DO NOT use sensor_fastcnt_irq */

    param);

} else {

/*

* turn off irq

*/

drv->os->gpio->irq_set(pin, POS_GPIO_MODE_INPUT_PU+POS_GPIO_MODE_IRQ_DISABLE, POS_NULL, POS_NULL);

if( s->drv_buf ) {

    drv->os->free(drv->s->drv_buf);

    s->drv_buf = POS_NULL;

}

}

return POS_STATUS_OK;

}

```

To facilitate RT Thread compilation and integration, the following LinkerScript.ld script can be used:

```
/* Entry Point */
```

```
ENTRY(main_start)
```

```
/* Highest address of the user mode stack */
```

```
_estack = 0xC0000038; /* end of RAM */
```

```
_Min_Heap_Size = 0; /* required amount of heap */
```

```
_Min_Stack_Size = 0; /* required amount of stack */
```

```
/* Memories definition */
```

```
MEMORY
```

```
{
```

```
    RAM (xrw)      : ORIGIN = 0xC0000000, LENGTH = 0x38
```

```
    ROM (rx)       : ORIGIN = 0x80000000, LENGTH = 24K
```

```
}
```

```
/* Sections */
```

```
SECTIONS
```

```
{
```

```
    /* The startup code into ROM memory */
```

```
    .hdr_vector :
```

```
    {
```

```
        . = ALIGN(4);
```

```
        KEEP(*(.hdr_vector)) /* Startup code */
```

```
        . = ALIGN(4);
```

```
    } >ROM
```

```
/* The program code and other data into ROM memory */
```

```
.text :
```

```
{
```

```
. = ALIGN(4);
```

```
*(.text)      /* .text sections (code) */
```

```
*(.text*)     /* .text* sections (code) */
```

```
*(.glue_7)    /* glue arm to thumb code */
```

```
*(.glue_7t)   /* glue thumb to arm code */
```

```
*(.eh_frame)
```

```
KEEP (*(init))
```

```
KEEP (*(fini))
```

```
. = ALIGN(4);
```

```
_etext = .;    /* define a global symbols at end of code */
```

```
} >ROM
```

```
/* Constant data into ROM memory*/
```

```
.rodata :
```

```
{
```

```
. = ALIGN(4);
```

```
*(.rodata)    /* .rodata sections (constants, strings, etc.) */
```

```
*(.rodata*)   /* .rodata* sections (constants, strings, etc.) */
```

```
. = ALIGN(4);
```

```
} >ROM
```

```
.ARM.extab : {  
  
    . = ALIGN(4);  
  
    *(.ARM.extab* .gnu.linkonce.armextab.*)  
  
    . = ALIGN(4);  
  
} >ROM
```

```
.ARM : {  
  
    . = ALIGN(4);  
  
    __exidx_start = .;  
  
    *(.ARM.exidx*)  
  
    __exidx_end = .;  
  
    . = ALIGN(4);  
  
} >ROM
```

```
.preinit_array :  
  
{  
  
    . = ALIGN(4);  
  
    PROVIDE_HIDDEN (__preinit_array_start = .);  
  
    KEEP (*( .preinit_array*))  
  
    PROVIDE_HIDDEN (__preinit_array_end = .);  
  
    . = ALIGN(4);  
  
} >ROM
```

```
.init_array :  
  
{  
  
    . = ALIGN(4);
```

```

PROVIDE_HIDDEN (__init_array_start = .);

KEEP (*(SORT(.init_array.*)))

KEEP (*(init_array*))

PROVIDE_HIDDEN (__init_array_end = .);

. = ALIGN(4);

} >ROM

```

```

.fini_array :

{

. = ALIGN(4);

PROVIDE_HIDDEN (__fini_array_start = .);

KEEP (*(SORT(.fini_array.*)))

KEEP (*(fini_array*))

PROVIDE_HIDDEN (__fini_array_end = .);

. = ALIGN(4);

} >ROM

```

```

/* Used by the startup to initialize data */

```

```

_sidata = LOADADDR(.data);

```

```

/* Initialized data sections into RAM memory */

```

```

.data :

{

. = ALIGN(4);

_sdata = .;    /* create a global symbol at data start */

*(.data)      /* .data sections */

```



```
*(.data*)    /* .data* sections */
```

```
. = ALIGN(4);
```

```
_edata = .;    /* define a global symbol at data end */
```

```
} >RAM AT> ROM
```

```
/* Uninitialized data section into RAM memory */
```

```
. = ALIGN(4);
```

```
.bss :
```

```
{
```

```
/* This is used by the startup in order to initialize the .bss section */
```

```
_sbss = .;    /* define a global symbol at bss start */
```

```
__bss_start__ = _sbss;
```

```
*(.bss)
```

```
*(.bss*)
```

```
*(COMMON)
```

```
. = ALIGN(4);
```

```
_ebss = .;    /* define a global symbol at bss end */
```

```
__bss_end__ = _ebss;
```

```
} >RAM
```

```
/* User_heap_stack section, used to check that there is enough RAM left */
```

```
._user_heap_stack :
```

```
{
```

```
. = ALIGN(8);
```

```

        PROVIDE ( end = . );

        PROVIDE ( _end = . );

        . = . + _Min_Heap_Size;

        . = . + _Min_Stack_Size;

        . = ALIGN(8);

    } >RAM


    /* Remove information from the compiler libraries */

    /DISCARD/ :

    {

        libc.a ( * )

        libm.a ( * )

        libgcc.a ( * )

    }

    .ARM.attributes 0 : { *(.ARM.attributes) }

}

```

3.4.3.3 Virtual and physical addresses

As the name implies, the virtual address itself is a non-existent address space, while the physical address is a real address space. The address is further divided into the ROM address where the program is stored and the RAM address where the memory is stored. Taking the current platform MCU series as an example, 0x00000000 is the real physical ROM address, 0x20000000 is the real physical RAM address, while 0x80000000 used by the application layer is a non-existent virtual ROM address, and 0xC0000000 is a non-existent virtual RAM address.

However, when a piece of code is compiled according to the virtual address, it must be located on a real physical address of the system when it finally runs. The virtual to physical address translation of some large CPU systems (typical Linux or Windows X86 platforms) is supported by the MMU module of the CPU, and this hardware mechanism ultimately provides a virtualized environment. However, in our micro system, the adoption of the CPU system supporting MMU will bring additional power consumption growth and obvious price cost disadvantage. Therefore, we can only simulate a similar virtual environment through software mechanism to realize a simulated memory and instruction virtualization.

3.4.3.4 Memory emulation(Data Address MMU)

Without the support of MMU physical module, the virtual address of instructions cannot be realized. We also use the form of simulation to simulate a similar system. Similar to the data memory, the instruction address space is also stored continuously, which can tell the compiler to use the relative offset position as the function access entrance during compilation, so as to avoid using absolute addresses for calls between functions during linking. However, when a function pointer is passed as a parameter, such as a timer or an interrupt callback function, it will cause an exception if it is passed to an interrupt to call a non-existent address, so some additional processing is required. The method is that the OS layer and the middle layer record the function mapping relationship, know the exact storage location (absolute physical address), and inform the application layer in the form of global variables. In this way, the application layer only needs to use the absolute physical address after translation mapping and pass it to each interrupt vector table or bottom callback entry as a callback function. Since all common functions will be declared in export, the bottom layer fully knows which functions and their virtual addresses are available, and then only needs to make correct mapping, thus realizing instruction simulation.

3.4.3.5 Driver file management

Because of the use of virtual addresses, driver files can be stored at will, independent of the running location, and their instructions and data addresses are not limited to storage, so the middle layer can easily manage driver files. We have divided a continuous address space in the physical space of the ROM to store each driver file, and the length information of the compiled driver file in the header can be used to locate the actual length of each driver. Therefore, we can achieve regular traversal and search by traversing each header sector. With OTA and CLI, the middle tier can add and delete single driver files that are common in all file systems, enabling flexible management of driver files.

3.5 Embedded CLI

3.5.1 Overview

By connecting the serial port to the debugging console of the device, you can use tools like SecureCRT or the Configuration Tool to conduct command line interaction (CLI) on the computer.

The CLI is divided into two levels:

- OS layer CLI: mainly short character commands to achieve the most basic version burning and initialization running status viewing
- Main task CLI (middle layer): It is mainly a long character command (typically in the form of four characters plus parameters), realizing almost all version upgrading, status viewing, configuration management, analog acquisition, module debugging and other operations

3.5.2 Serial port console

By default, the serial port adopts 9600 baud rate, 8 data bits, 1 stop bit, 0 check bit. Multi character commands need to be input at one time without pausing. On the SecureCRT terminal, one-time input can be realized by using the command interaction window at the bottom, or similar input can be completed by using the Configuration Tool.

3.5.3 Operation Guide

The system startup serial port will automatically print the version status information. At this time, you can enter short commands such as 0~9 in the prompt to intervene. After about 5 seconds, the system will load the main task and provide the long command operation capability through the CLI of the main task (the short command is still valid). When the CLI stops operation, the system will automatically enter a low-power sleep state according to the scheduling cycle. After that, once the next cycle window arrives or there is any CLI operation on the serial port, the system will wake up and immediately respond to the CLI operation. The following are all supported CLI commands.

Short command (provided by OS kernel):

- Command: 0 (system reset)

The system will reset after prompting for confirmation and pressing Y

- Command: 1 (Help)

Print all short command help

- Command: 2 (view status information)

Display current OS running information, including stack memory size, system running time and sleep time

- Command: 3 (Erase all ROM and only keep OS)

After prompting for confirmation and pressing Y, the system will erase all drives and applications, including the main application. When an incorrect driver or main application is downloaded, you can use this command to erase and only retain the OS.

- Command: 4 (upgrade OS)

Upgrade the OS version through the serial port.

- Command: 5 (Upgrade the main application)

Upgrade the main application (M-server version) through the serial port.

- Command: 6 (upgrade drive)

Upgrade the all-in-one driver package (including private applications) through the serial port.

- Command: 7/8/9/(internal reserved)

Do not operate at will.

Long command (provided by the main application):

- Command: help

The main task help displays a list of all long commands. You can view the supported dbg debugging commands, pin mapping functions and configuration file parameter items through help dbg/help pin/help set.

- Command: cfgp (pin mapping display)

Displays the current IO pin mapping configuration.

- Command: cfgr (configuration file view)

Displays the contents of the current profile.

- Command: del XXXX (delete a drive)

Delete a drive. XXXX is the name of the drive (you can view it with the list command) or the type of drive (for example, 2 temperature sensors). XXXX can also be *. At this time, all application layer drivers and custom applications will be deleted, leaving only the OS and the main application. The current configuration is not affected by the command.

- Command: dump

The current configuration is displayed with detailed explanation. All 0 parameters without configuration will also be displayed.

- Command: list (view all drive applications)

View all driver details, including name, version, and occupancy size. With the help of list/del/xupg, it can realize the comprehensive upgrade, deletion, inspection and other management of drive applications.

- Command: srun (restore the system to normal operation)

When the system is in the stopped state, normal operation can be resumed through this command. This command is used together with the stop command to suspend the collection report and resume it.

- Command: stop (restore the system to normal operation)

When the system is in the stopped state, normal operation can be resumed through this command. This command is used together with the srun command to suspend the collection report and resume it.

- Command: show

The current configuration is displayed with detailed explanation. All 0 parameters without configuration will not be displayed.

- Command: set XXXX VVVVV (modify configuration parameters)

Modify the configuration item XXXX to the parameter value VVVVV. The content of the configuration item can be viewed with `cfg/show/dump`.

- Command: stop

Suspend the system to stop it. This command is used together with the `srun` command to suspend the collection report and resume it.

- Command: xupg (serial port upgrade)

Upgrade a file through serial port XMODEM. The modified file can be pin configuration file, system configuration file, OS kernel version, main application version, driver package file, single driver version, etc.

- Command: info (system status view)

View the detailed system status, including the kernel OS version, main application version, stack status, system operation and other information, as well as the status of some major modules.

- Command: adc XXX (ADC sampling)

Use the specified pin to sample ADC and display the sampling value. XXX can be `pa0/pa1/.../pf0` and other pin names.

- Command: inp XXX (view IO pin input status)

View the current input status (0/1) of the specified pin. XXX can be `pa0/pa1/.../pf0` and other pin names.

- Command: mod XXX VVVV (IO pin mode modification)

Modify the working mode of the specified pin. XXX can be `pa0/pa1/.../pf0` and other pin names. VVVV is the corresponding working mode. See POS/SDK online manual for details.

- Command: out XXX VVVV (IO pin output control)

Controls the output height (0/1) of the specified pin. XXX can be pin names such as `pa0/pa1/.../pf0`. VVVV is 0-low and 1-high. The pin needs to be set to OUTPUT mode in advance.

- Command: pin XXX VVVV (IO pin mode modification)

Modify the pin mapping, XXX is the pin function viewed by `cfgp`, and VVVV is the pin group.

- Command: pins

View the current mode and working status of all pins.

- Command: dbg XXX (debug mode)

Internal debugging command, see help dbg for details.

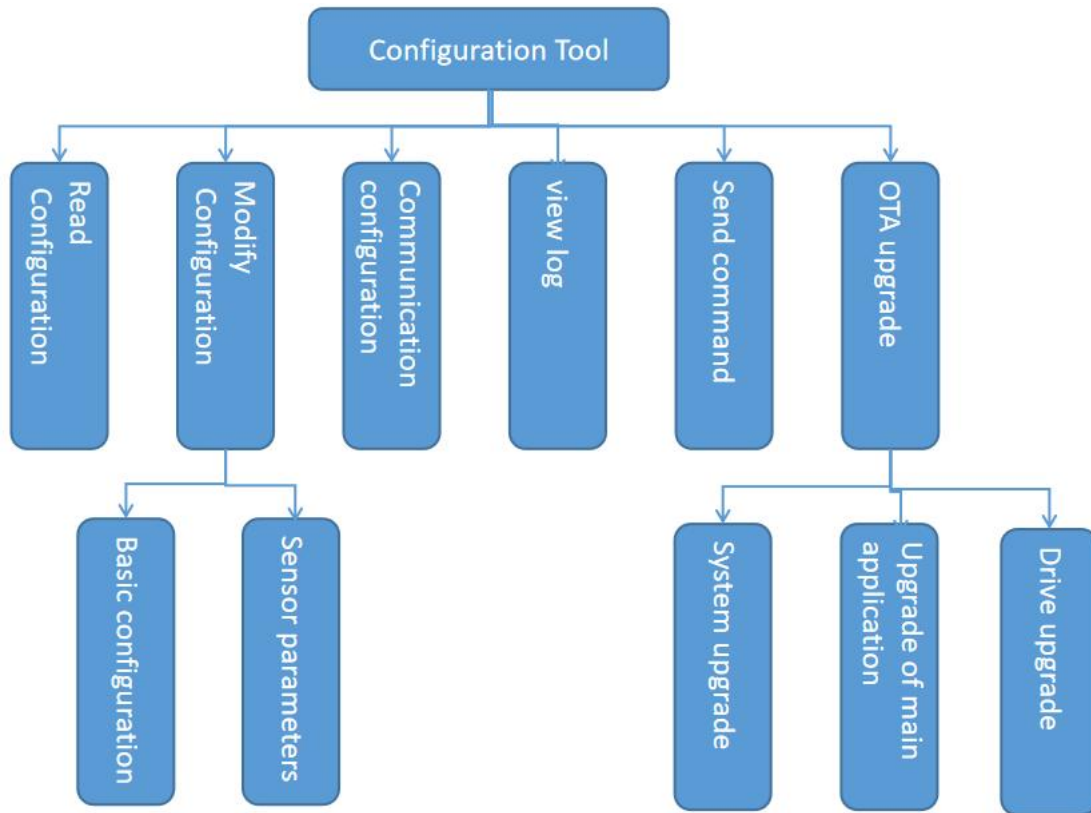
3.6 Configuration Tool

3.6.1 Overview

WxS series products adopt a new architecture design, which enhances the universality and expandability of the products. The Configuration Tool software specially used for product configuration adjustment has been developed to make the product configuration easier. The main characteristics are as follows:

- Supports up to eight sensors
- Sleep power consumption as low as 3uA
- More sophisticated power management, turn off on demand
- Application dynamic loading and management, which can be updated through the cloud
- Host computer configuration and parameter adjustment, more convenient to use
- The communication mode can be changed at will, and one device only supports one
- Support multiple communication protocols
- The format of reported data is flexible, extensible and user-defined
- Built in multiple exception handling mechanisms to ensure the stable operation of the product
- Scalable security architecture
- There are sensor application packages supporting hundreds of sensors
- Support edge computing, organically combine sensors and controllers

3.6.2 Functional framework



The overall function of the Configuration Tool is shown in the figure above, which can read the configuration parameters and display them in the interface, modify the basic configuration, sensor parameters and other configuration information, and burn the configuration information through the Save Configuration button. At the same time, OTA upgrade is supported, and system upgrade, main application upgrade, drive upgrade and other operations can be performed. Refer to Section 3.7.2 for upgrade methods.

3.6.3 Installation Guide

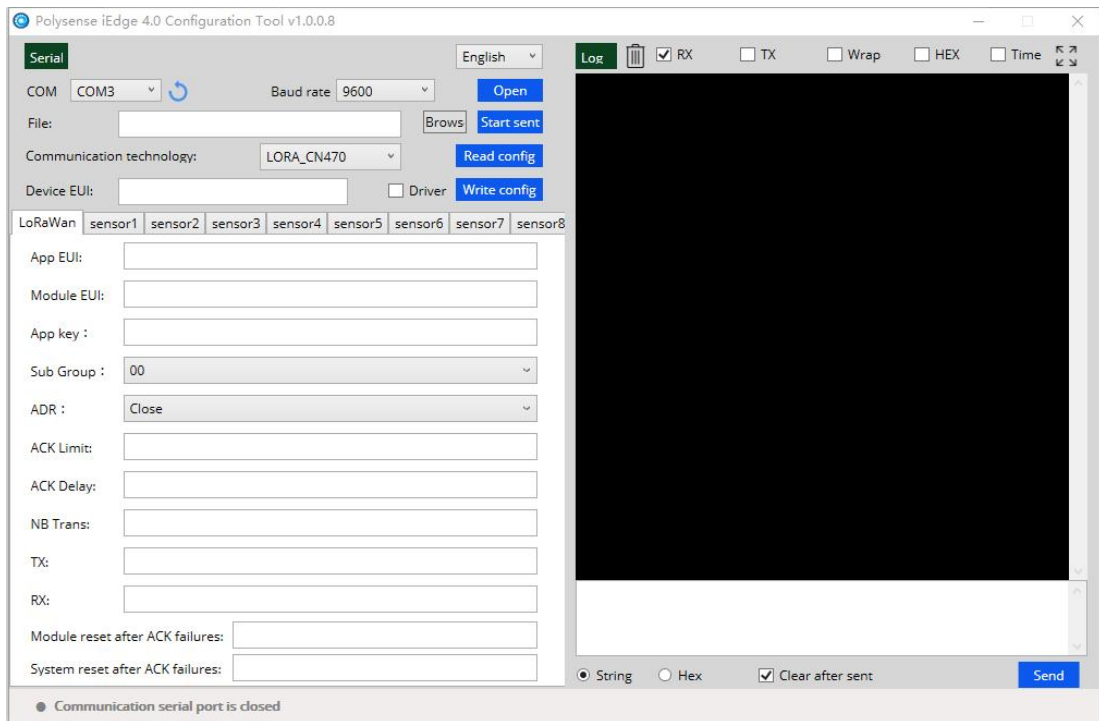
- ❖ The operation of the Configuration Tool requires the installation of a dependent environment.NET Framework 4.0

Configuration Tool download address:<http://ota.polysense.online/wincc/ConfigurationTool.rar>

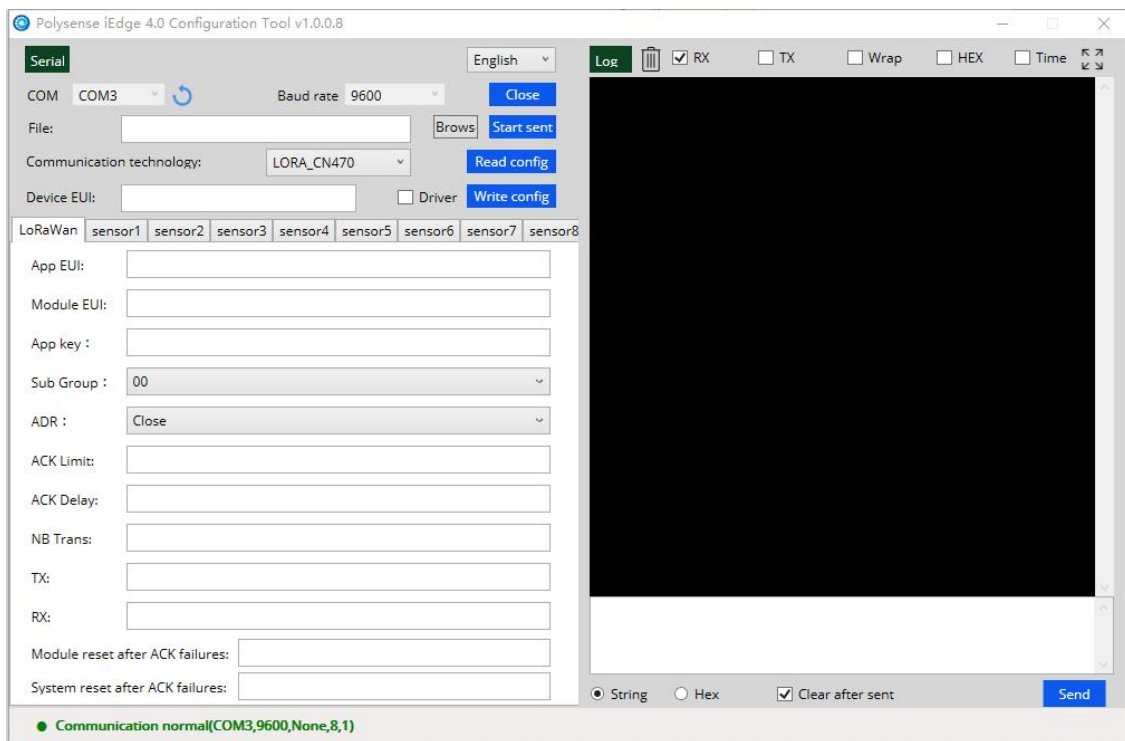
Dependency package download address:<http://ota.polysense.online/wincc/depend/dotNet4.rar>

Driver download address: <http://ota.polysense.online/iEdge4.0/>

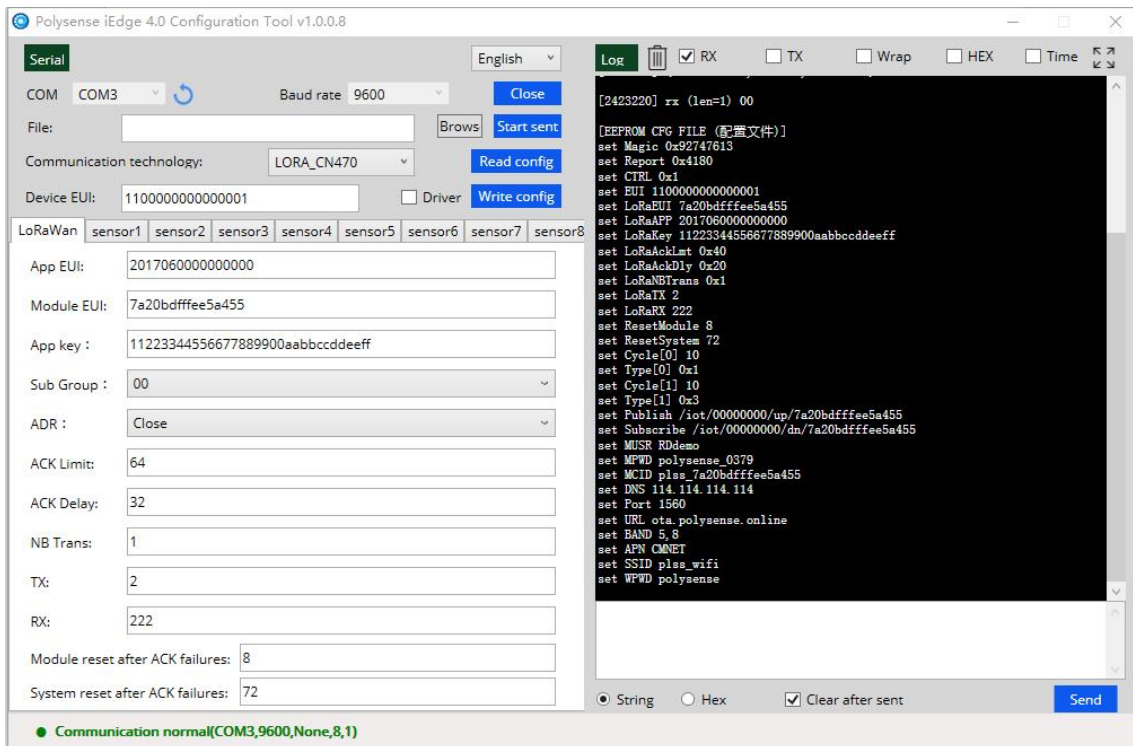
- ❖ Download the software, double-click to run directly, and the interface is as follows



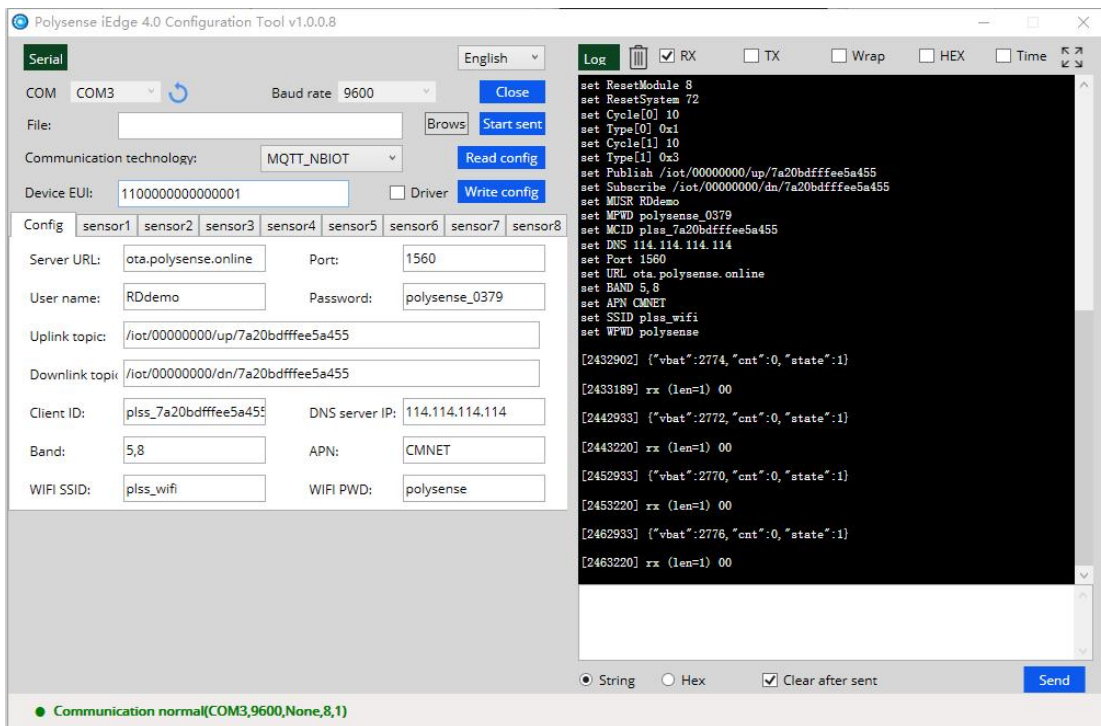
- ❖ After selecting language, serial port and baud rate (9600 by default), click to open serial port



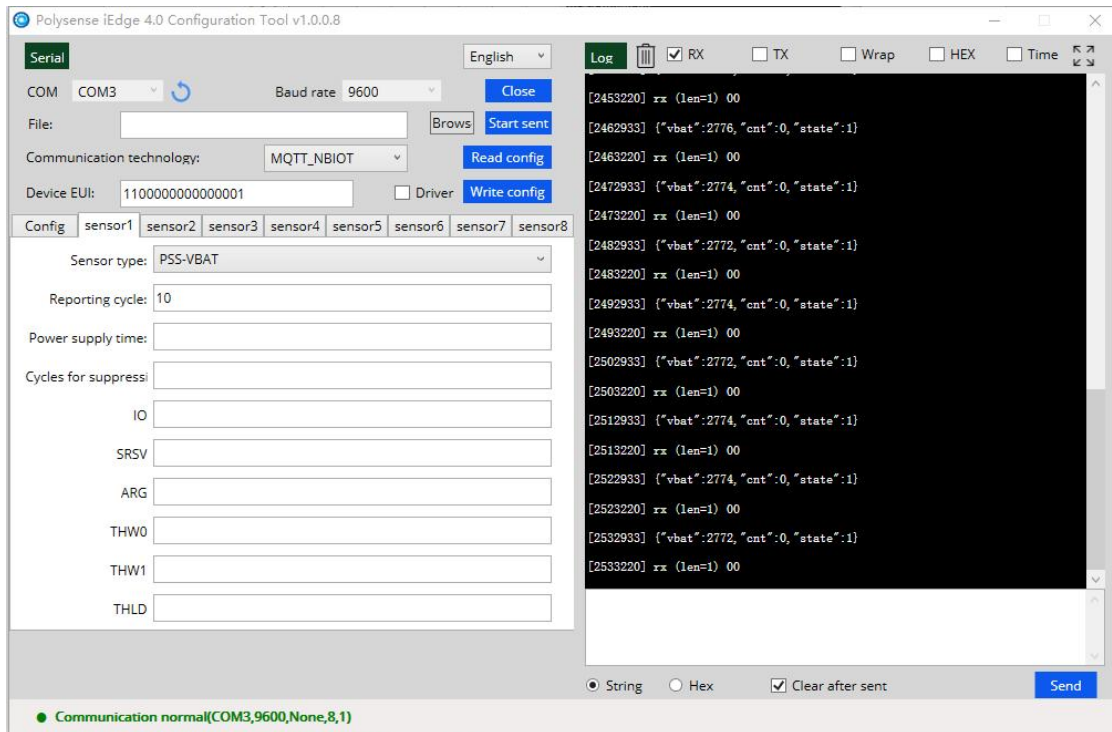
- ❖ Click to read the configuration



The content to be configured is different for different communication types. The following figure shows MQTT_NBIOT, at this time, you need to configure the server URL, port, user name, password, reporting subject, receiving subject, etc.:



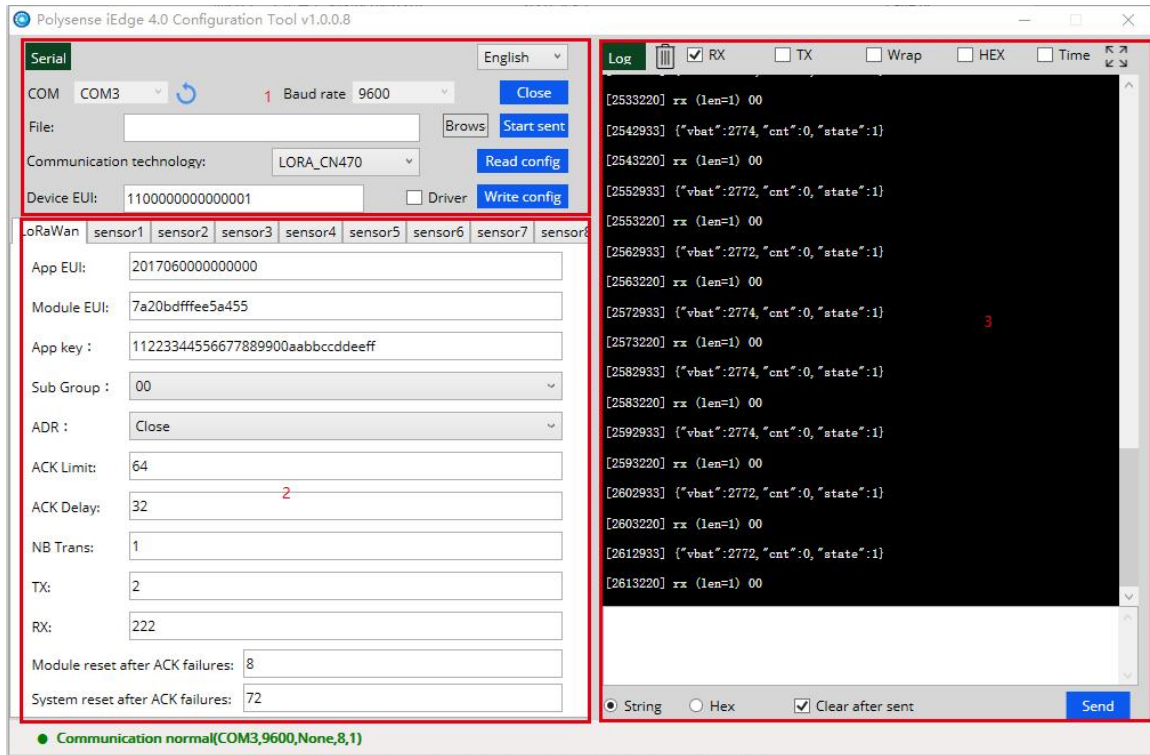
- ❖ Select the specific sensor type according to the hardware wiring situation and demand, configure the relevant information of the reporting cycle, power supply time and suppression reporting cycle



- ❖ Make sure that the device is connected correctly, then click the 'Save Configuration' button to send the configuration information to the device end, and see the prompt "Transfer completed...." on the Configuration Tool's status bar, It means that the configuration is successful and the device can be reset to operate normally.

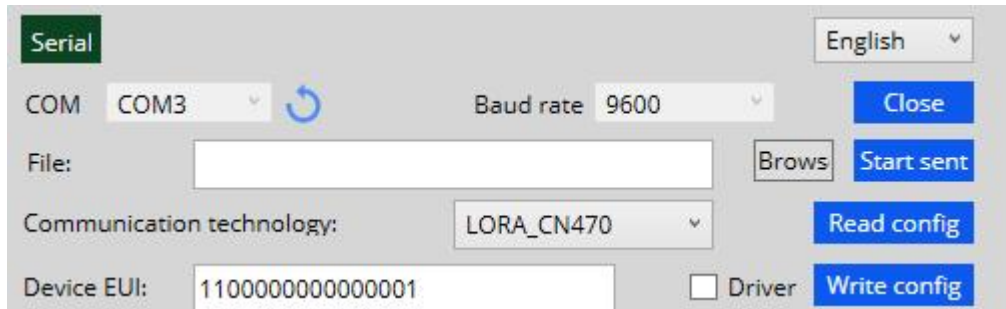
3.6.4 Operation Guide

Polysense intelligent IoT terminal Configuration Tool is a support configuration tool specially developed for WxS series products, which aims to provide customers with the ability to adjust products conveniently and quickly. The interface after opening is as follows:



The area 1 is the general function area, the area 2 is the parameter adjustment area, and the area 3 is the device operation information debugging area.

General functions



- ❖ Language selection: the Configuration Tool design supports multiple languages, and the required language can be selected from the drop-down list
- ❖ Serial port: supports automatic search of serial ports. When connecting multiple serial ports, the first one is selected by default
- ❖ Baud rate: select the corresponding baud rate according to the need, default 9600
- ❖ Open serial port: set and open the serial port according to the selected serial port information
- ❖ Brows: select a single file to be burned as required, mainly for development and test
- ❖ Start sent: send the currently selected file to the selected download address
- ❖ Read config: read the configuration information of the connected equipment
- ❖ Save config: send the currently selected configuration information to the device

- ❖ Driver: When checked, the corresponding drive of the sensor will be automatically burned when the configuration is saved

Parameter adjustment

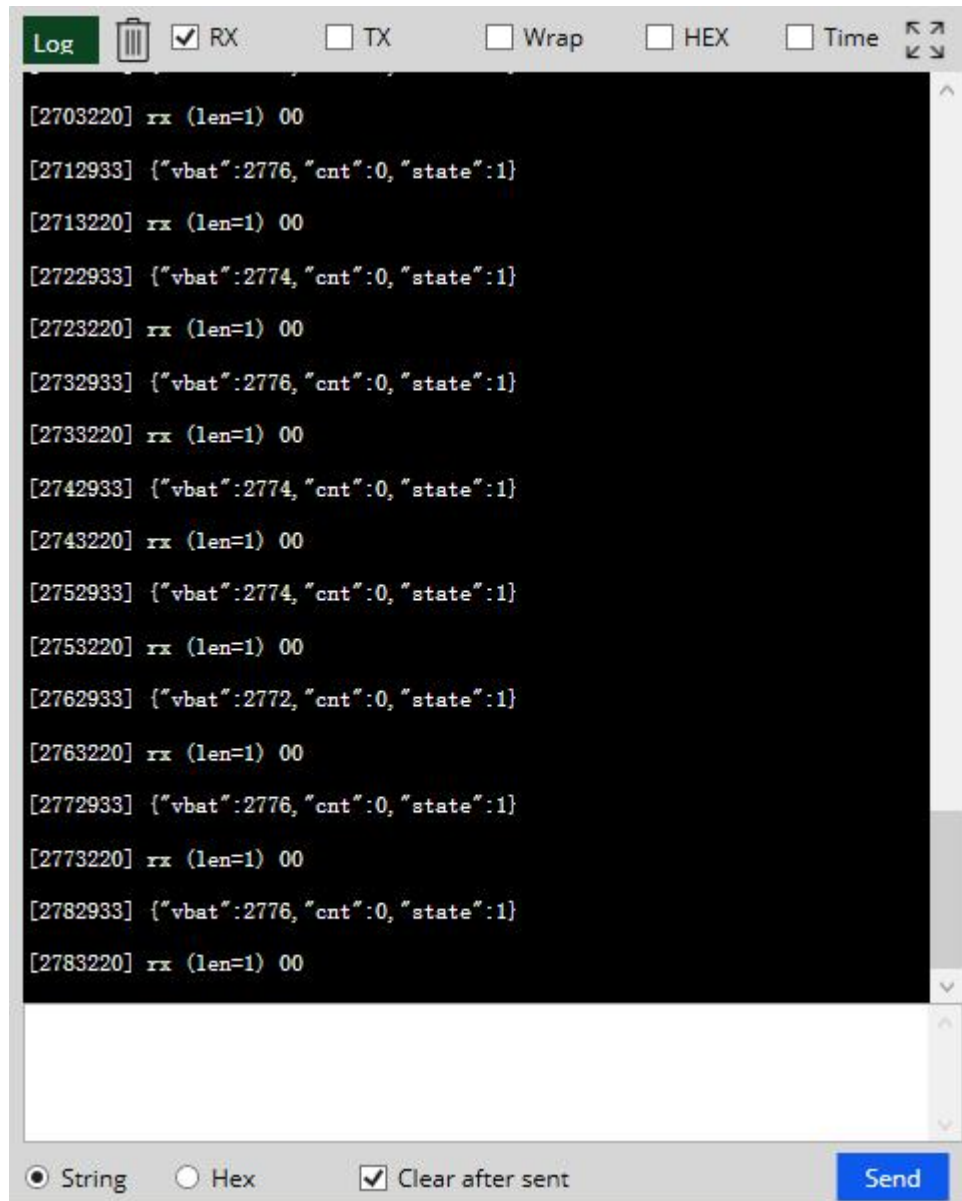
This area contains all adjustable parameters, which are mainly divided into LoRaWan (or Config, different tabs are displayed for different communication types), 8 sensor configurations, and all parameters can pop up parameter interpretation information after hovering the mouse. The specific parameter interpretation information can be viewed by opening the software:

LoRaWan	sensor1	sensor2	sensor3	sensor4	sensor5	sensor6	sensor7	sensor8
App EUI:	<input type="text" value="2017060000000000"/>							
Module EUI:	<input type="text" value="7a20bdfffee5a455"/>							
App key :	<input type="text" value="11223344556677889900aabbccddeeff"/>							
Sub Group :	<input type="text" value="00"/>							
ADR :	<input type="text" value="Close"/>							
ACK Limit:	<input type="text" value="64"/>							
ACK Delay:	<input type="text" value="32"/>							
NB Trans:	<input type="text" value="1"/>							
TX:	<input type="text" value="2"/>							
RX:	<input type="text" value="222"/>							
Module reset after ACK failures:	<input type="text" value="8"/>							
System reset after ACK failures:	<input type="text" value="72"/>							

Run ang debug

You can view the operation of the device in this area, send specific instructions, and debug and adjust the equipment.

You can view all supported detailed instruction rules by sending instruction help.



```

Log [X] RX [ ] TX [ ] Wrap [ ] HEX [ ] Time [ ]
[2703220] rx (len=1) 00
[2712933] {"vbat":2776,"cnt":0,"state":1}
[2713220] rx (len=1) 00
[2722933] {"vbat":2774,"cnt":0,"state":1}
[2723220] rx (len=1) 00
[2732933] {"vbat":2776,"cnt":0,"state":1}
[2733220] rx (len=1) 00
[2742933] {"vbat":2774,"cnt":0,"state":1}
[2743220] rx (len=1) 00
[2752933] {"vbat":2774,"cnt":0,"state":1}
[2753220] rx (len=1) 00
[2762933] {"vbat":2772,"cnt":0,"state":1}
[2763220] rx (len=1) 00
[2772933] {"vbat":2776,"cnt":0,"state":1}
[2773220] rx (len=1) 00
[2782933] {"vbat":2776,"cnt":0,"state":1}
[2783220] rx (len=1) 00

String Hex [X] Clear after sent [X] Send
  
```

3.7 OTA firmware update

3.7.1 Overview

The upgrade data of all software can be upgraded through the Configuration Tool in addition to the serial port tool. Through the Configuration Tool embedded in the cloud, the version of the driver database can be verified and upgraded online. The required drivers (communication drivers, sensor drivers) can be loaded dynamically, and the required drivers can be loaded as light as possible. The unnecessary drivers need not be loaded, which better releases the computing power of the edge end, and reduces the power consumption of the device and the consumption of resources.

3.7.2 OTA implementation mode

- ❖ Input the command xupg in the [Run debugging window], click ‘Send’, and the equipment terminal will enter the upgrade mode.
- ❖ Select the upgraded drive file in the [General function area], click the [Start sent] button, and the progress bar will display the upgrade progress, and wait for the upgrade to complete.

Driver download address:<http://ota.polysense.online/iEdge4.0/>

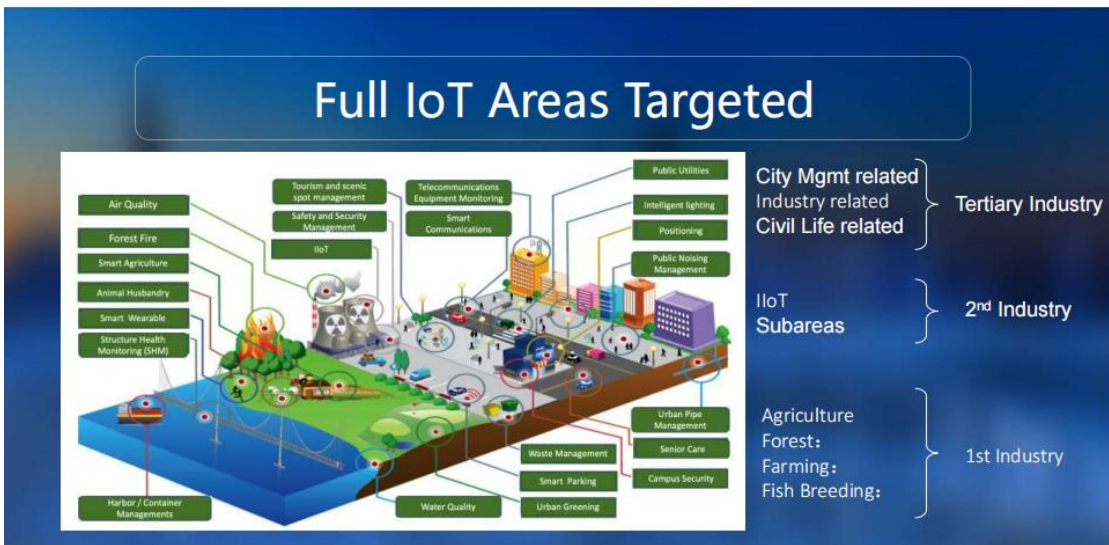
The terminal can use all serial port tools that support xmodem protocol to upgrade, such as terarm, securecr... The operation steps are the same as that of the Configuration Tool. First send the xupg command, then select and upload the file.

4. Value-Added Product Development

4.1 Third party integrated development library document

To facilitate users of professional technical teams or third-party integrators to carry out new development work on the basis of existing Polysense products, please refer to the latest development library documents in the following links:<http://ota.polysense.online/iEdge4.0/doc/files.html>

5.Product application



5.1 Primary industry

The industries involved include agriculture, forestry, animal husbandry, aquaculture and fishery.

No.	Monitoring programme	Application scenario	Sensor type
-----	----------------------	----------------------	-------------

1、 Agriculture			
1.1	Soil moisture monitoring	Used for agricultural soil environment monitoring	Temperature, humidity, PH, conductivity, moisture content
1.2	Monitoring of vegetable greenhouses	Used to detect the growth environment of vegetable greenhouse crops	Light intensity, CO ₂ , temperature and humidity, water content, nitrogen, phosphorus, potassium, UV
1.3	Agrometeorological monitoring	Used for agrometeorological weather detection	Air pressure, temperature and humidity, wind speed, wind direction, rainfall, light and total radiation
1.4	Warehouse granary monitoring	Used for granary storage inspection	Temperature, humidity, dust
1.5	Mushroom bag monitoring	Used for temperature and humidity detection of mushroom bags	Temperature and humidity probe
1.6	Seeder controller	Used for sowing control of fertilizers and seeds	RTU control unit
2、 Forestry			
2.1	Forest fire prevention	Used for forest fire detection	CO, temperature, smoke detection
2.2	Air quality in forest area	Used for air quality in forest area	Negative oxygen ions, temperature and humidity, AQI, PM2.5, noise
3、 Animal husbandry			
3.1	Health&Location Monitoring	Used for animal safety testing in animal husbandry	Electronic fence, movement amount and position
3.2	Growth environment inspection	Animal growth environment detection in animal husbandry industry	Temperature and humidity, CO ₂ , hydrogen sulfide, ammonia, noise, dust
4、 Sideline			
4.1	Cold chain transportation	Used for leading transportation chain inspection	Temperature, humidity, position, track
4.2	Traffic vehicle tracking	Used for vehicle tracking and location information	GPS
5、 Fisheries			
5.1	Growth environment monitoring	Detection of growth environment of freshwater and seawater aquaculture	Water temperature, conductivity, PH value, turbidity, dissolved oxygen, heavy metals, ammonia nitrogen, transparency, ORP, TOD, fluoride, residual chlorine
5.2	Water quality monitoring	Water quality monitoring	

5.2 Secondary industry

The industries involved include industry, brewing industry, electrolytic aluminum industry, steel industry, etc.

No.	Monitoring programme	Application scenario	Sensor type
1.1	Equipment temperature monitoring and	Temperature warning for smelting and processing industry	Suction cup temperature measurement+PAD display

	warning		
1.2	Temperature measurement of electrolytic aluminum production line	Temperature measuring system for electrolytic aluminum plant	High temperature probe+platform
1.3	Temperature monitoring in brewing industry	Used for temperature detection in liquor making and koji making workshops	Integrated temperature measurement products
1.4	Workshop noise management	Used for noise detection in heavy industries such as processing and manufacturing	Noise sensor
1.5	Product count detection	Product count for intensive production	Counting sensor
1.6	Mine belt tear detection	Used for tear detection and early warning of mine conveyor belt	Tear detection sensor+idler temperature+camera
1.7	Intelligent detection of chemical plant	Industrial park and production inspection for chemical industry	18 kinds of gas detection, temperature, smoke detection and intrusion sensors
1.8	Preventive machine maintenance	Used to detect the vibration frequency, amplitude and temperature of equipment	Vibration, temperature and power supply status
1.9	Water and electric meter reading	Used for metering and reading of water and gas meters	Water electric flow sensor
1.10	Pipeline leakage monitoring	For leak monitoring of transmission pipeline	Flow meter, temperature and leakage sensor
1.11	Distribution box monitoring	Used for monitoring internal parameters of distribution box/cabinet	Current, voltage, inert gas, gate magnet, immersion
1.12	Industrial perimeter security monitoring	For perimeter intrusion prevention and security monitoring	PIR, microwave radar, camera
1.13	Fire monitoring in steel plant	For safety monitoring of fire pool workshop in steel plant	Water immersion sensor, smoke detector, temperature
1.14	asset management	Asset management for important equipment	GPS, RFID, three-axis

5.3 Tertiary industry

The industries involved include urban environment, smart parks, fire safety, safety precautions, property energy consumption management, health care, etc.

No.	Monitoring programme	Application scenario	Sensor type
1、 Metropolitan environment			
1.1	Outdoor multi-function environment microstation	Widely used in various scenarios of environmental standard monitoring	Ambient temperature and humidity, light pressure, wind direction/speed, rainfall, noise, etc

	detection		
1.2	Soil environment detection	Widely used to detect the soil environment in the city	Soil temperature, humidity, PH value, EC value, etc
1.3	AQI outdoor microstation detection	Widely used in urban areas, factories, enterprises, streets, schools, hospitals and other environmental monitoring areas	AQI (PM2.5, PM10, SO2, NO2, CO, O3)
1.4	Indoor air purification test	Widely used in shopping malls, office buildings, homes and other places	TVOC, PM2.5, CO2, CH2O (formaldehyde), temperature and humidity (more than ten other gases can be expanded as required)
1.5	Emission pollution detection of catering industry	Widely used in smoke emission points such as restaurants in the city and restaurants along the street	Temperature measurement of oil smoke sensor and kitchen smoke exhaust pipe
2、 Smart Park			
2.1	Parking lot status detection	It is widely used to detect the occupancy of parking spaces in parking lots, automobile exhaust, combustible gas, etc	Ultrasonic distance sensor, combustible gas sensor (including natural gas, biogas, propane, butane, acetylene, hydrogen, etc.)
2.2	Toilet status detection	Widely used in toilet occupancy and smoking monitoring	IR, two-way passenger flow statistics, H2S, NH3, smoking sensor, PM10
2.3	Inspection of well cover and well position	It is widely used in urban streets, factories, schools, hospitals and other public areas for well cover opening and theft monitoring, detection of underground combustible gas, underground water level, etc	Well cover limit switch sensor, CO, CH4 and water level sensor can be matched with various sensing functions
2.4	Garbage bin status detection	Widely used in the detection of garbage cans being stolen and garbage spillage in public areas such as urban areas, streets, schools and hospitals	Ultrasonic distance sensor, temperature probe, tilt switch
2.5	Pole tilt impact	Widely used for monitoring the inclination, toppling and impact of city streets and light poles inside the city	(ADIS16,209, LIS2DH) Inclination sensor, 2-axis/3-axis acceleration sensor
2.6	Public equipment box safety monitoring	It is widely used for opening monitoring of fire equipment boxes, public equipment boxes and toolboxes in indoor, corridor and urban areas	PIR+magnetic door switch, ambient temperature and humidity
2.7	Roof water tank status monitoring	It is widely used for water tank condition monitoring of various building clusters in the city	PIR+door magnetic switch, ambient temperature and humidity, water level, water temperature
3、 Fire safety			
3.1	Building structure safety monitoring	Widely used for monitoring the inclination, settlement and cracks of buildings in urban areas	Inclination, crack, laser ranging
3.2	Building Security	It is widely used to monitor	IR, door magnet, laser

	Intrusion Monitoring	whether there is intrusion inside the buildings in the city	intrusion protection
3.3	Monitoring of power room	It is widely used for monitoring the power room in various buildings in the city, including environmental monitoring and preventive maintenance of equipment in the power room	Adhesive temperature and sound (1-1.5m); Current transformer, ambient temperature and humidity; Water leakage (battery solution): external water leakage, ambient temperature and humidity/attached temperature (or individual water leakage products)
4. Security system			
4.1	Indoor positioning of important materials	It is widely used in the positioning of important hospital equipment, public assets of the company, warehouse materials, etc	RFID, 2-axis/3-axis acceleration sensor, material displacement switch
4.2	Outdoor positioning of vehicle materials	It is widely used for vehicle occupancy and positioning in urban areas; Outdoor positioning detection of important materials	GPS, 3-axis
4.3	Comprehensive detection of data center	Widely used for water leakage and electric leakage detection in important machine rooms, data centers, warehouses and other places in the city	Water leakage sensor, current transformer (2-4 sensors driven by one), ambient temperature and humidity, attached temperature measurement (6-8 sensors driven by one), noise
4.4	Comprehensive detection of urban fire protection and security	It is widely used to detect the lower limit of smoke detection and flammable and explosive gas in urban smoking areas, important machine rooms, underground warehouses and other areas	Smoke sensing, lower explosion limit of flammable gas (including natural gas, biogas, propane, butane, acetylene, hydrogen), CO, temperature and humidity
4.5	Urban public health emergency response plan	Apply to similar	Infrared temperature measurement, face recognition
5. Property energy consumption management system			
5.1	Payment management of water and electricity meters	Widely used in urban water and electricity meter reading and payment detection	Custom development
5.2	Statistical analysis of property energy consumption	Widely used in intelligent remote meter reading; Household electricity meter, electricity diversion of shared housing, electricity metering of electrical equipment, electricity consumption of shopping malls/factories	Intelligent energy consumption meter/customized development
5.3	Temporary electricity supervision	It is widely used in detection of temporary power consumption, overload and power balance in communities, urban areas,	Current transformer, intelligent energy consumption meter

households, factories, etc			
6、 Healthy elderly care			
6.1	Safety protection	Safety protection for the elderly at home	Three axis, GPS, door magnet, emergency switch, gas leakage, smoke detector, fire, water immersion, PIR
7、 Others			
7.1	Campus temperature measurement pedestrian flow	Used for campus temperature measurement and pedestrian flow detection	Temperature measurement PAD
7.2	Monitoring of energy power plant	For power plant industry	CO, NH3, temperature, PIR, microwave radar, voltage, current, tilt
7.3	Shop fire	Used for pre-warning of catering and store fire	CO, temperature, CH4
7.4	retail	Location information and business conditions for chain stores	GPS, power consumption, counting
7.5	Vaccine cold chain	For vaccine storage management	Temperature and humidity alarm
7.6	Epidemic isolatio	Used in the scene of epidemic home isolation	Door magnet, one call switch