SIM7070G Arduino NB-IoT / LTE / GNSS / GPRS / GPS Expansion

Overview

The NB-IoT expansion board, designed specifically for Arduino controllers, is globally frequency-compatible, supporting CAT-M, NB-IoT, GSM, GPRS, EDGE communication methods, and GNSS satellite positioning functions.

Its ultra-low power consumption and precise positioning characteristics, coupled with support for CAT-M and GNSS satellite positioning, make it an ideal choice for smart logistics, asset tracking, smart cities, and other fields.

This expansion board is not only suitable for prototype development but also for small batch production, making it an optimal choice for low-power, low-latency, and medium-throughput applications. It has built-in GNSS satellite positioning, including GPS, GLONASS, Galileo, and BD, making it highly suited for IoT applications such as remote control and mobile tracking.

The SIM7070G also comes with a rich set of built-in AT commands and provides detailed libraries, user manuals, and application documents, including usage methods for network communication protocols such as HTTP, MQTT, FTP, CoAP, NIDD, PING, etc., enabling quick start without deep understanding of underlying network protocols.

Order Code

Order Code	Brand	Description
E06002-001	DFRobot	SIM7070G Arduino NB-IoT / LTE / GNSS / GPRS / GPS Expansion Shield (Global)

Overview

Most importantly, the SIM7070G has passed relevant certifications in multiple countries and regions worldwide, specifically:

- Anatel: Brazil Telecommunication Equipment Certification
- AT&T: AT&T Certification
- CCC: China Compulsory Certification
- CE: European CE Certification
- UKCA: UK Conformity Assessed Certification
- FCC: Federal Communications Commission Certification
- GCF: Global Certification Forum
- IC: Canadian Telecommunication Equipment Certification
- JATE: Japanese Industrial Standards Certification
- TA: China Radio Transmission Certification
- NCC: Taiwan Telecommunications Terminal Certification
- KC: Korea Certification
- PTCRB: PTCRB Certification Organization
- RCM: Australia New Zealand Compliance Certification

Overview

- REACH: European Chemicals Certification
- RoHS: European Environmental Certification
- TELEC: Japanese Telecommunication Equipment Certification
- T-mobile: T-mobile Mobile Communication Service Provider Certification
- US Cellular: US Cellular Mobile Communication Service Provider Certification
- Verizon: Verizon Mobile Communication Service Provider Certification
- Deutsche Telekom: German Telecommunication Certification

SIM7070G Arduino NB-IoT / LTE / GNSS / GPRS / GPS Expansion

Features

- Supports global frequency bands for NB-IoT
- Supports CAT-M communication
- Supports GNSS satellite positioning (GPS, CLONASS, Galileo, BD)
- Supports multiple frequency bands including CAT-NB/CAT-M/GSM/GPRS/EDGE
- Comes with built-in AT commands, software library is encapsulated, eliminating the need for lower-level development
- Comprehensive certifications

Technical Specification

•Operating Voltage: 5V (Note: The module will consume a large amount of current at the moment of network connection, an external power supply is needed during operation)

•Input Voltage: 7~12VDC

•Data Transmission

- GPRS: 85.6Kbps(UL), 107Kbps(DL)
- EDGE: 296Kbps(UL), 236.8Kbps(DL)
- LTE Category M1: 589Kbps (DL)
- LTE Category M1: 1119Kbps (UL)
- LTE Category NB1/NB2: 127Kbps (DL)
- LTE Category NB1/NB2: 158.5Kbps (UL)
- EGPRS: Compatible with MCS 12

Technical Specification

- Supported Bands
  - NBIoT
    - B1 B2 B3 B4 B5 B8 B12 B13 B18 B19 B20 B25 B26 B28 B66 B71 B85
  - CAT-M (LTE-HD-FDD)
    - B1 B2 B3 B4 B5 B8 B12 B13 B14 B18 B19 B20 B25 B26 B27 B28 B66 B85
  - GSM/GPRS
    - GSM850MHz
  - EDGE
    - EGSM900MHz
    - DCS1800MHz
    - PCS1900MHz

Technical Specification

- GNSS
  - GPS
  - GLONASS
  - BeiDou
  - Galileo
  - Protocol: NMEA

•Interfaces

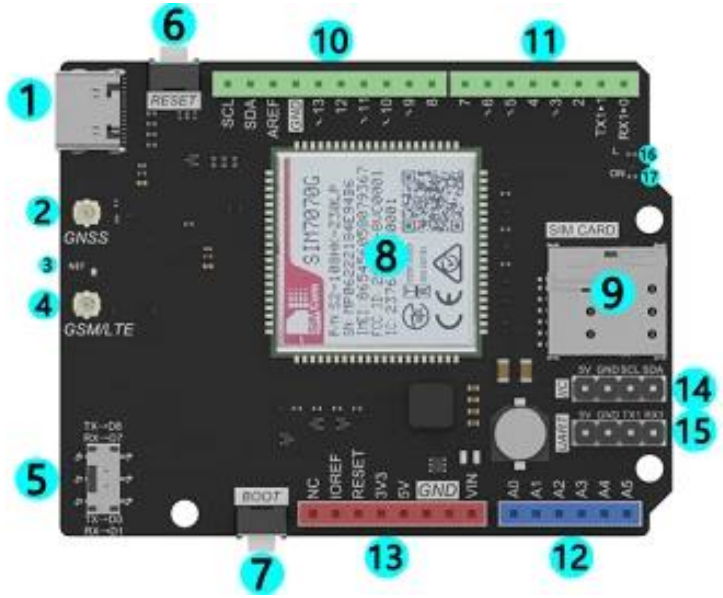
- NBIoT Antenna Interface: IPEX1
- GNSS Antenna Interface: IPEX1
- SIM Card: NBIoT/CAT-M/2G Card (Only supports 1.8V SIM card, not 3V card)

Function Diagram

Please Note:

1.Software/Hardware Serial Switch: Since controllers such as Arduino UNO/Mega default D0 and D1 as hardware serial ports connected to the USB port, many expansion boards that require the use of serial ports may encounter serial port conflicts during use. Therefore, a software serial port interface is specially designed, allowing users to communicate with Arduino using the software serial port. For extended reading:

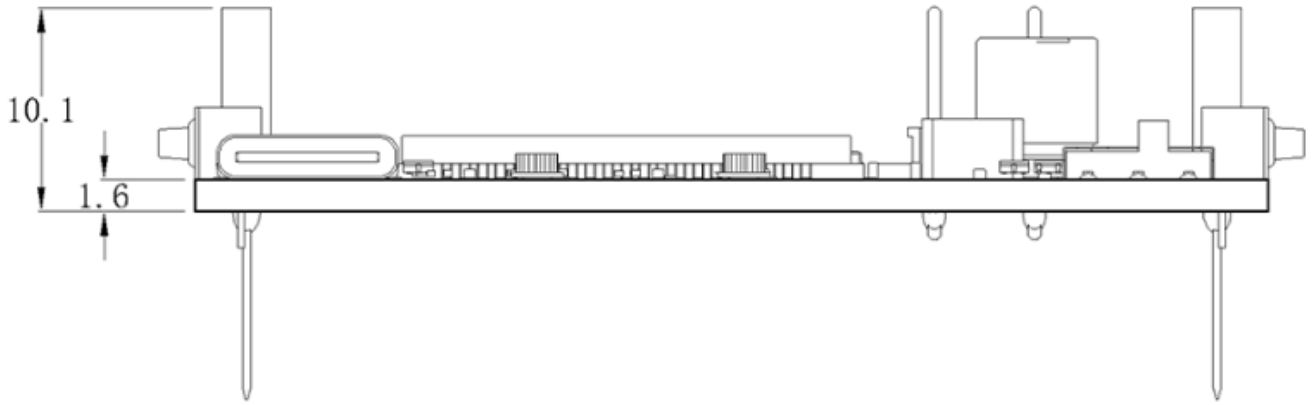
2.For convenient program control, the Boot button is by default connected to the D12 pin. The module can be powered on by pulling the D12 pin high for 2 seconds. After powering on, the SIM7070G will have an initialization time of about 2 seconds. After initialization, it can be used normally.



1. USB Firmware Update Interface
2. GNSS Antenna
3. Network Indicator
4. GSM/LTE Antenna
5. Software/Harware Serial Port Switch
6. Arduino Reset
7. SIM7070G Control Button
8. SIM7070G Module
9. SIM Socket
10. Arduino Digital Interface
11. Arduino Digital Interface
12. Arduino Analog Interface
13. Arduino Power Supply Interface
14. Arduino IIC Interface
15. Arduino UART Interface
16. Arduino Blink Indicator
17. Power Supply Indicator

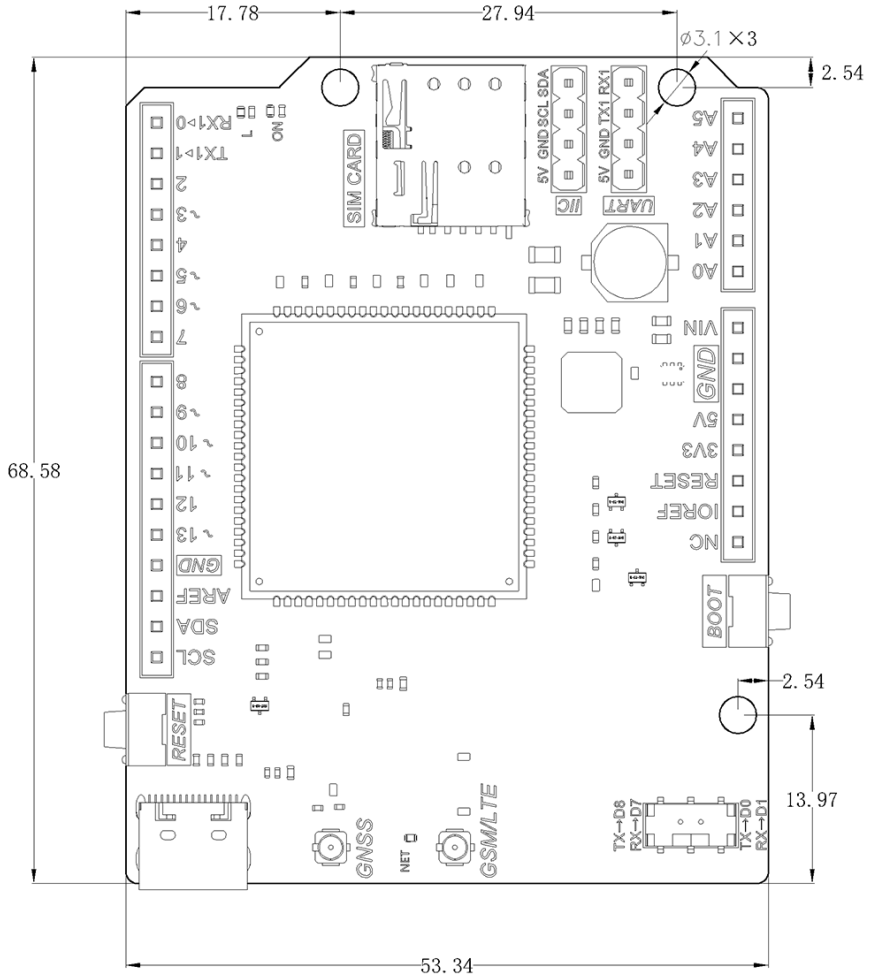
Dimension

Unit: mm



Dimension

Unit: mm



Tutorial (Based on SIM7070G Library)

Requirements

•Hardware

- DFR0216 DFRduino UNO R3 - Arduino Compatible x1
- SIM7070G Arduino NB-IoT/LTE/GPRS Expansion Shield x1
- USB Wire x1
- 7V~12V DC Power Supply x1

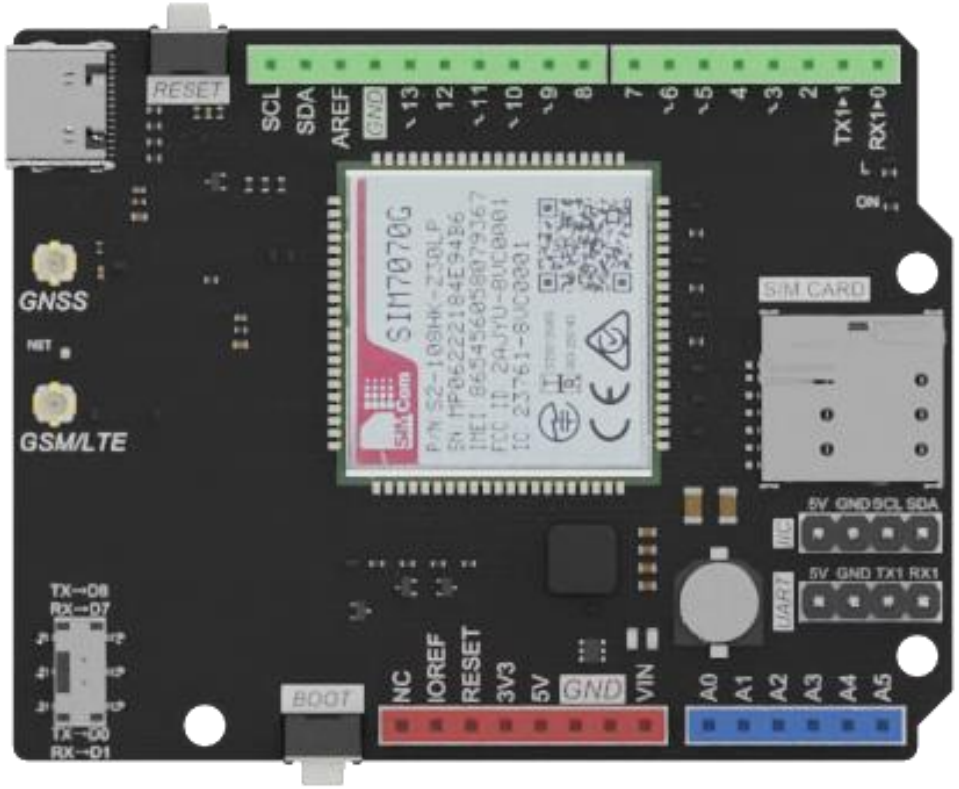
•Software

- Install drivers: Download the driver file and decompress it on the desktop, connect the expansion board to the computer, and install five drivers in the Device Manager.
- Arduino IDE Click to Download Arduino IDE from Arduino®
- Please click to download [DFRobot General SIM Library]
- Please click to download [NB-IoT SIM7000 Shield Library] How to install Libraries in Arduino IDE.

SIM7070G Arduino NB-IoT / LTE / GNSS / GPRS / GPS Expansion

Tutorial (Based on SIM7070G Library)

Plug the SIM7070G Arduino NB-IoT/LTE/GPRS Expansion Shield to DFRduino UNO R3 directly.
NOTE: The module SIM7070G requires a 7-12V external power supply. And please switch the hardware and software serial control switch to TX-D8, RX-D7.



Sample Code (Serial AT Command)

- Since Arduino UNO only has one hardware serial port, it is recommended to use software serial communication when using it. The SIM7000 library file uses software serial by default. Please switch the software/hardware serial control switch to TX>D8, RX>D7.
- This example uses serial port to send AT commands to control the SIM7070G expansion board.

Results

- Control the SIM7070G expansion board by sending AT commands in the serial monitor and observe the response of the expansion board.

```
#include <DFRobot_SIM7070G.h>

#define PIN_TX     7
#define PIN_RX     8
SoftwareSerial     mySerial(PIN_RX, PIN_TX);
DFRobot_SIM7070G    sim7070g(&mySerial);

void setup()
{
  delay(1500);
  Serial.begin(19200);
  mySerial.begin(19200);

  Serial.println("Turn ON SIM7070G......");
  if (sim7070g.turnON()) {
    Serial.println("Turn ON !");
  }
```

Sample Code (Serial AT Command)

```
  Serial.println("Set baud rate.....");
  while (1) {
    if (sim7070g.setBaudRate(19200)) {
      Serial.println("Set baud rate:19200");
      break;
    } else {
      Serial.println("Faile to set baud rate");
      delay(1000);
    }
  }

  Serial.println("For example, if you type AT\\r\\n, OK\\r\\n will be responsed!");
  Serial.println("Enter your AT command :");
}

void loop()
{
  mySerial.listen();
  while (mySerial.available()) {
    Serial.write(mySerial.read());
  }
  mySerial.flush();
  while (Serial.available()) {
    mySerial.write(Serial.read());
  }
}
```

Sample Code (GNSS Positioning)

- When using GNSS positioning, first plug in the GPS antenna and place the expansion board outdoors.
- Results
- The serial monitor will print some successful initialization information, followed by latitude and longitude information.

```
#include <DFRobot_SIM7070G.h>

#define PIN_TX    7
#define PIN_RX    8
SoftwareSerial    mySerial(PIN_RX, PIN_TX);
DFRobot_SIM7070G    sim7070g(&mySerial);

void setup()
{
  delay(1500);
  Serial.begin(115200);
  mySerial.begin(19200);

  Serial.println("Turn ON SIM7070G......");
  if (sim7070g.turnON()) {
    Serial.println("Turn ON !");
  }

  Serial.println("Set baud rate......");
  while (1) {
    if (sim7070g.setBaudRate(19200)) {
      Serial.println("Set baud rate:19200");
      break;
    } else {
```

Sample Code (GNSS Positioning)

```
Serial.println("Faile to set baud rate");
    delay(1000);
   }
  }

  Serial.println("Check SIM card.....");
  if (sim7070g.checkSIMStatus()) {
   Serial.println("SIM card READY");
  } else {
   Serial.println("SIM card ERROR, Check if you have insert SIM card and restart SIM7070G");
   while (1);
  }

  Serial.println("Init positioning function.....");
  while (1) {
   if (sim7070g.initPos()) {
    Serial.println("Positioning function initialized");
    break;
   } else {
    Serial.println("Fail to init positioning function");
    delay(1000);
   }
  }
}
```

Sample Code (GNSS Positioning)

```
void loop()
{
  Serial.println("Enter anything end with CRLF to get positioning ");
  char loge[10];
  readSerial(loge);
  Serial.println("Getting position.....");
  if (sim7070g.getPosition()) {
    Serial.print(" Longitude : ");
    Serial.println(sim7070g.getLongitude());
    Serial.print(" Latitude : ");
    Serial.println(sim7070g.getLatitude());
  } else {
    Serial.println("Wrong data try again");
  }
}
```

Sample Code (GNSS Positioning)

```
int readSerial(char result[])
{
  int i = 0;
  while (1) {
    while (Serial.available() > 0) {
      char inChar = Serial.read();
      if (inChar == '\n') {
        result[i] = '\0';
        Serial.flush();
        return 0;
      }
      if (inChar != '\r') {
        result[i] = inChar;
        i++;
      }
    }
  }
}
```

Sample Code (MQTT Connection)

- This example operates the SIM7070G expansion board to perform MQTT communication.
- Results
- The serial monitor will print some information, read the data you input, and send the data to the specified topic.

```
#include <DFRobot_SIM7070G.h>

#define serverIP       "iot.dfrobot.com"
#define IOT_USERNAME    "USERNAME"
#define IOT_KEY        "PASSWORD"
#define IOT_TOPIC      "TOPIC"
#define IOT_CLIENT     "dfrobot"
#define PIN_TX         7
#define PIN_RX         8

SoftwareSerial         mySerial(PIN_RX, PIN_TX);
DFRobot_SIM7070G       sim7070g(&mySerial);

void setup()
{
  delay(1500);
  int signalStrength;
  Serial.begin(115200);
  mySerial.begin(19200);

  Serial.println("Turn ON SIM7070G.....");
  if (sim7070g.turnON()) {
    Serial.println("Turn ON !");
  }
```

Sample Code (MQTT Connection)

```
Serial.println("Set baud rate.....");
while (1) {
  if (sim7070g.setBaudRate(19200)) {
    Serial.println("Set baud rate:19200");
    break;
  } else {
    Serial.println("Faile to set baud rate");
    delay(1000);
  }
}
Serial.println("Check SIM card.....");
if (sim7070g.checkSIMStatus()) {
  Serial.println("SIM card READY");
} else {
  Serial.println("SIM card ERROR, Check if you have insert SIM card and restart SIM7070G");
  while (1);
}

Serial.println("Set net mode.....");
while (1) {
  if (sim7070g.setNetMode(sim7070g.eNB)) {
    Serial.println("Set NB mode");
    break;
  } else {
    Serial.println("Fail to set mode");
    delay(1000);
  }
}
```

Sample Code (MQTT Connection)

```
  Serial.println("Get signal quality.....");
  delay(1500);
  signalStrength = sim7070g.checkSignalQuality();
  Serial.print("signalStrength =");
  Serial.println(signalStrength);
  delay(500);

  Serial.println("Attaching service.....");
  while (1) {
    if (sim7070g.attacthService()) {
      Serial.println("Attach service");
      break;
    } else {
      Serial.println("Fail to Attach service");
      delay(1000);
    }
  }
}
```

Sample Code (MQTT Connection)

```
void loop()
{
 String  sendData;
 Serial.print("Connect to :");
 Serial.println(serverIP);
 if (sim7070g.openNetwork(sim7070g.eTCP, serverIP, 1883)) {              //Connect to server
  Serial.println("Connected !");
 } else {
  Serial.println("Failed to connect");
  return;
 }
 delay(200);

 Serial.print("Connect to : ");
 Serial.println(IOT_USERNAME);
 if (sim7070g.mqttConnect(IOT_CLIENT, IOT_USERNAME, IOT_KEY)) {
  Serial.println("Connected !");
 } else {
  Serial.println("Failed to connect");
  return;
 }
 delay(200);
 Serial.println("Input data end with CRLF : ");
 sendData = readSerial(sendData);
 Serial.print("Send data : ");
 Serial.print(sendData);
 Serial.println(" ......");
```

Sample Code (MQTT Connection)

```
if (sim7070g.mqttPublish(IOT_TOPIC, sendData)) {
   Serial.println("Send OK");
  } else {
   Serial.println("Failed to send");
   return;
  }
  delay(200);

  Serial.println("Close connection......");
  if (sim7070g.closeNetwork()) {
   Serial.println("Close connection !");
  } else {
   Serial.println("Fail to close connection !");
   return;
  }
  delay(2000);
}
```

Sample Code (MQTT Connection)

```
String readSerial(String result)
{
  int i = 0;
  while (1) {
    while (Serial.available() > 0) {
      char inChar = Serial.read();
      if (inChar == '\n') {
        result += '\0';
        while (Serial.read() >= 0);
        return result;
      }
      if (i == 50) {
        Serial.println("The data is too long");
        result += '\0';
        while (Serial.read() >= 0);
        return result;
      }
      if (inChar != '\r') {
        result += inChar;
        i++;
      }
    }
  }
}
```

API function

```
/**
 * @fn DFRobot_SIM7070G
 * @brief DFRobot_SIMcore constructor of abstract class. Construct serial ports
 * @param s The pointer to abstract class, where you can fill in the pointer to serial object.
 * @return None
 */
DFRobot_SIM7070G(Stream *s);
~DFRobot_SIM7070G(){};

/**
 * @fn recv
 * @brief Receive
 * @param buf Receive data content
 * @param maxlen Receive data length
 * @return Get data length
 */
int recv(char* buf, int maxlen);
```

API function

```
/**
 * @fn checkSignalQuality
 * @brief Check signal quality
 * @return 0-30:Signal quality
 */
int checkSignalQuality(void);

/**
 * @fn batteryPower
 * @brief Battery power
 * @return Battery power
 */
int batteryPower(void);

/**
 * @fn setNetMode
 * @brief Set net mode
 * @param net The net mode
 * @n    GPRS: GPRS mode
 * @n    NB:   NB-IOT mode
 * @return bool type, indicating the status of setting
 * @retval ture Success
 * @retval false Failed
 */
bool setNetMode(eNet net);
```

API function

```
/**
 * @fn attacthService
 * @brief Open the connection
 * @return bool type, indicating the status of opening the connection
 * @retval true Success
 * @retval false Failed
 */
bool attacthService(void);

/**
 * @fn setBaudRate
 * @brief Set baud rate to avoid garbled
 * @param rate Baud rate value
 * @n    Possible values:1200 2400 4800 9600 19200 38400
 * @note SIM7070G default baud rate is 115200, reduce the baud rate to avoid distortion
 * @return bool type, indicating the status of setting
 * @retval true Success
 * @retval false Failed
 */
bool setBaudRate(long rate);
```

API function

```
/**
 * @fn checkSIMStatus
 * @brief Check SIM card
 * @return bool type, indicating the status of checking SIM card
 * @retval true Success
 * @retval false Failed
 */
bool checkSIMStatus(void);

/**
 * @fn openNetwork
 * @brief Start up connection
 * @param ptl  Choose connection protocol
 * @n    TCP  Choose TCP
 * @n    UDP  Choose UDP
 * @param host Host domain name
 * @param port  Contented port
 * @return bool type, indicating the status of opening Network
 * @retval true Success
 * @retval false Failed
 */
bool openNetwork(eProtocol ptl, const char *host, uint16_t port);
```

API function

```
/**
 * @fn closeNetwork
 * @brief End the connection
 * @return bool type, indicating the status of closing Network
 * @retval true Success
 * @retval false Failed
 */
bool closeNetwork(void);

/**
 * @fn turnON
 * @brief Turn ON SIM7070G
 * @return bool type, indicating the status of turning on
 * @retval true Success
 * @retval false Failed
 */
bool turnON(void);

/**
 * @fn initPos
 * @brief Init SIM7070G positioning module
 * @return bool type, indicating the initialization status
 * @retval true Success
 * @retval false Failed
 */
bool initPos(void);
```

API function

```
/**
 * @fn mqttConnect
 * @brief MQTT connect request
 * @param iot_client Client name user-defined
 * @param iot_username The user name identifies the name of the user who is connecting
 * @param iot_key The password for user
 * @return bool type, indicating the connection status
 * @retval true Success
 * @retval false Failed
 */
bool mqttConnect(const char* iot_client, const char* iot_username, const char* iot_key);

/**
 * @fn mqttPublish
 * @brief MQTT send command
 * @param iot_topic Target topic
 * @param iot_data  The data you want to send
 * @return bool type, indicating status of sending
 * @retval true Success
 * @retval false Failed
 */
bool mqttPublish(const char* iot_topic, String iot_data);
```

API function

```
/**
 * @fn mqttSubscribe
 * @brief Subscribe MQTT channel
 * @param iot_topic The subscribed MQTT key
 * @return bool type, indicating subscription status
 * @retval true Success
 * @retval false Failed
 */
bool mqttSubscribe(const char* iot_topic);

/**
 * @fn mqttUnsubscribe
 * @brief Unsubscribe MQTT channel
 * @param iot_topic The unsubscribed MQTT key
 * @return bool type, indicating unsubscribe status
 * @retval true Success
 * @retval false Failed
 */
bool mqttUnsubscribe(const char* iot_topic);
```

API function

```
/**
 * @fn mqttRecv
 * @brief MQTT send data
 * @param iot_topic Subscribe channel key
 * @param buf Send data
 * @param maxlen Send data length
 * @return bool type, indicating subscription status
 * @retval true Success
 * @retval false Failed
 */
bool mqttRecv(char* iot_topic, char* buf,int maxlen);

/**
 * @fn mqttDisconnect
 * @brief MQTT disconnection
 * @return bool type, indicating disconnection status
 * @retval true Success
 * @retval false Failed
 */
bool mqttDisconnect(void);
```

API function

```
/**
 * @fn httpInit
 * @brief Initialize HTTP service
 * @param net The net mode
 * @n    eGPRS: GPRS mode
 * @n    eNB:   NB-IOT mode
 * @return bool type, indicating initialization status
 * @retval true Success
 * @retval false Failed
 */
bool httpInit(eNet net);

/**
 * @fn httpConnect
 * @brief Connect to server
 * @param host Server IP
 * @return bool type, indicating connection status
 * @retval true Success
 * @retval false Failed
 */
bool httpConnect(const char *host);
```

API function

```
/**
 * @fn httpPost
 * @brief HTTP POST
 * @param host URL
 * @param data POST data
 * @return bool type, indicating request status
 * @retval true Success
 * @retval false Failed
 */
bool httpPost(const char *host , String data);

/**
 * @fn httpGet
 * @brief HTTP GET This function print the get data
 * @param host URL
 */
void httpGet(const char *host);

/**
 * @fn httpDisconnect
 * @brief Disconnect from server and cancel initialization
 */
void httpDisconnect(void);
```

API function

```
/**
 * @fn send
 * @brief Send data with specify the length
 * @param buf The buffer for data to be sent
 * @param len The length of data to be sent
 * @return bool type, indicating status of sending
 * @retval true Success
 * @retval false Failed
 */
bool send(char *buf,size_t len);

/**
 * @fn send
 * @brief Send data
 * @param data The data to send
 * @return bool type, indicating status of sending
 * @retval true Success
 * @retval false Failed
 */
bool send(char *data);
```

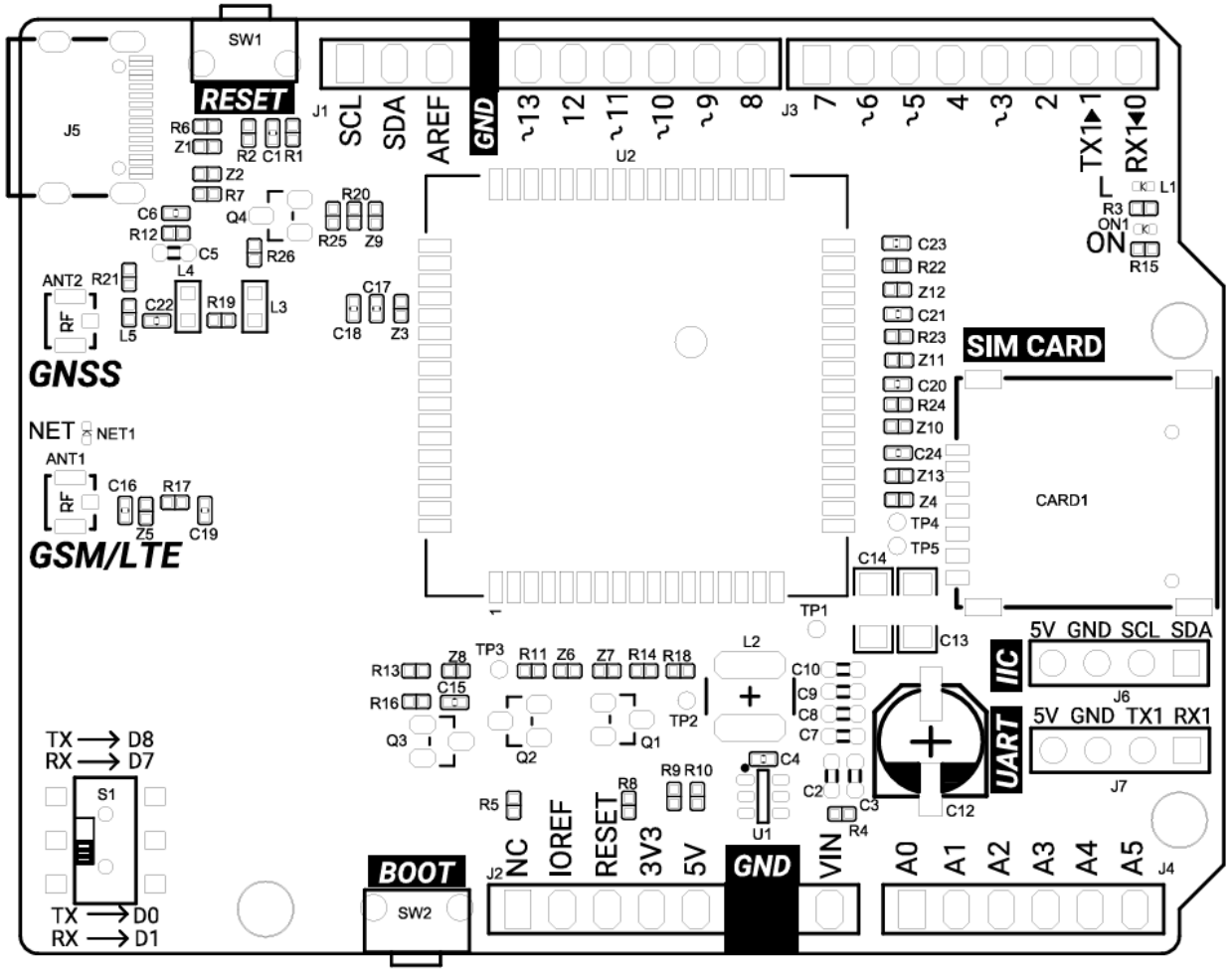
API function

```
/**
 * @fn getPosition
 * @brief Get the current position
 * @return bool type, indicating the status of getting position
 * @retval true Success
 * @retval false Failed
 */
bool getPosition(void);

/**
 * @fn getLatitude
 * @brief Get latitude
 * @return Latitude value
 */
const char* getLatitude(void);

/**
 * @fn getLongitude
 * @brief Get longitude
 * @return Longitude value
 */
const char* getLongitude(void);
```

Layout



PCB Information	Name:DFR0572 V2.1.0	Thickness: 1.6mm
Engineer:HDJ-CD	Top SolderMask: Black	Bottom SolderMask: Black
Date: 20240606	Top Silkscreen: White	Bottom Silkscreen: White

FAQ

Q: After burning the sample code, the serial port output cannot be initialized?

A: (1) Check whether the external power supply is 7~12V, powered through the DC2.1 black round hole on the main board, or through the vin, gnd port on the expansion board. (2) Check whether two antennas are installed. (3) Check the TX>D8, RX>D7 dip switch on the expansion board. (4) See if the red power indicator light on the expansion board is always on, and the NET indicator light is blinking. (5) Check whether the NB card is inserted. After checking, press the Boot button to restart and observe.

Q: Can SIM7070G use communication and GNSS positioning at the same time?

A: SIM7070G cannot use communication and GNSS positioning functions at the same time.

For more questions and interesting applications, you can **visit the forum** for reference or to post.

Documents and Files

- SIM7070G-AT Command Manual V1.01
- SIM7070G-2D_DXF
- SIM7070G-3D_STP

Revision History

Date	Revision	Change description
30/10/2025	1.0	Initial release