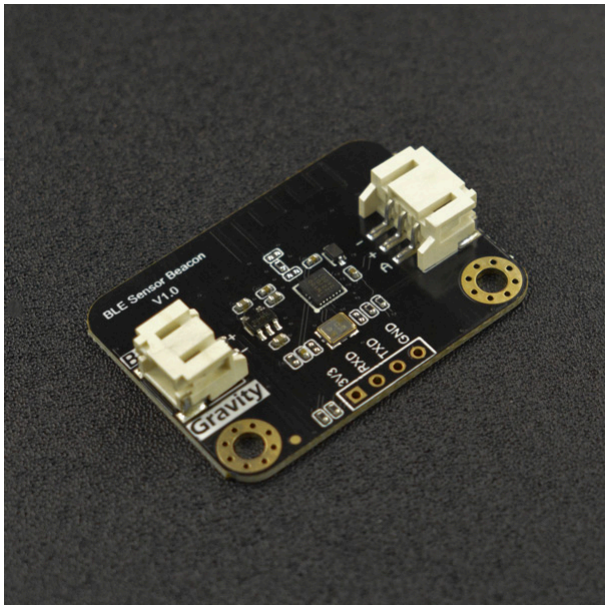
## SKU:DFR1103 (https://www.dfrobot.com.cn/goods-3635.html)



(https://www.dfrobot.com.cn/goods-3635.html)




## Introduction

This product is a module with integrated GNSS and RTC chips. It can receive various satellite signals such as BeiDou and GPS to obtain accurate time information and calibrate the time for RTC to ensure high accuracy and stability of time. It provides users with a simple and convenient way to calibrate and maintain the time of the device, and is suitable for various application scenarios that require precise time synchronisation.

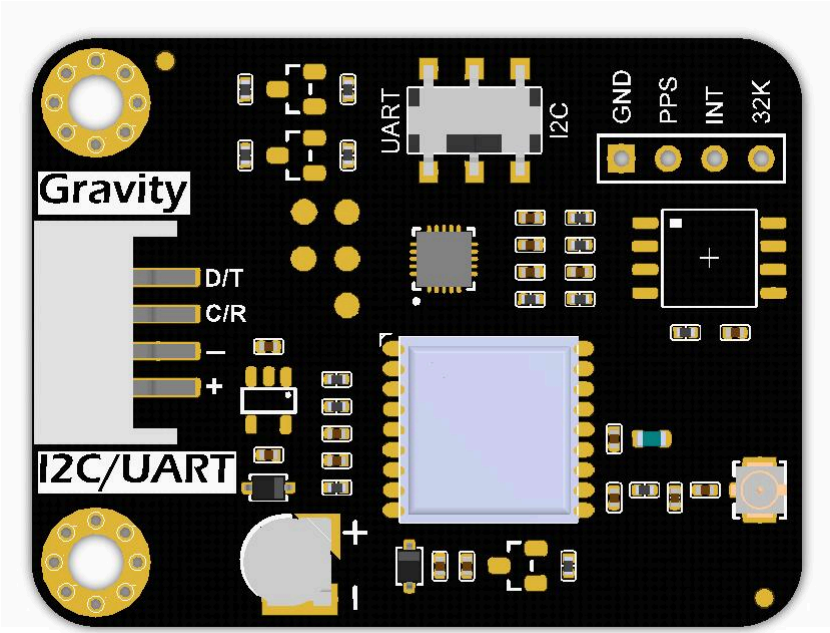
In cases where GNSS signals are not available, the onboard RTC chip can be used to obtain the time. In outdoor scenarios where low power consumption is required, the API can also be used to cut off the power supply to the GNSS to significantly reduce power consumption.

## Specification

- Operation Voltage： 3.3V~5V DC
- Output Signal：I2C/UART
- GNSS Accuracy：2.0m CEP
- First positioning time: Cold start 30S, Hot start 2S

- Operation current: 46mA (GNSS chip on), <1mA (GNSS chip off). The above are all tested under 5V condition.
- Antenna Interface: IPEX 1
- PCB Size：32mm * 42mm

## Pinout



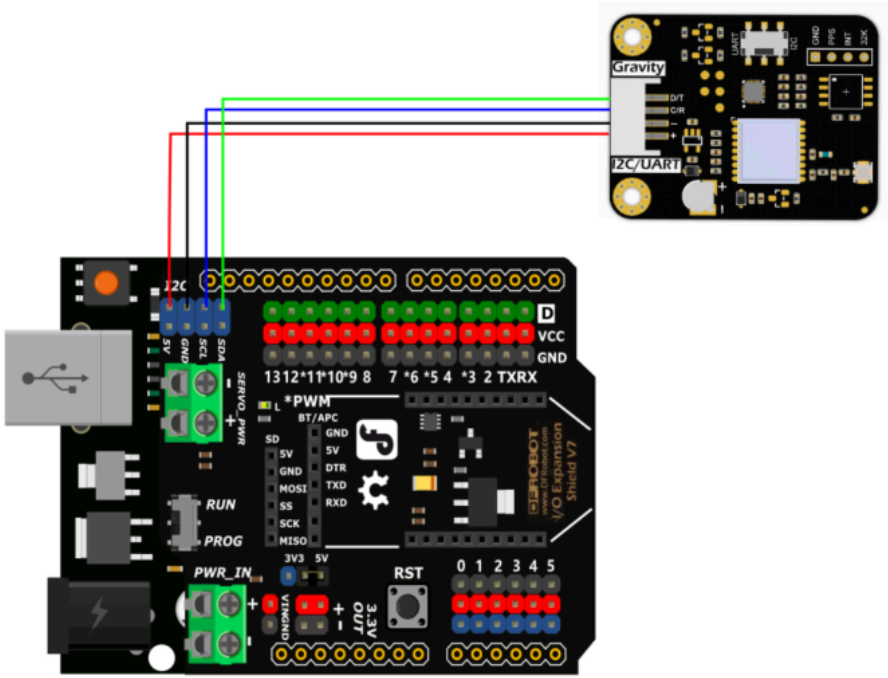

| Num | Label | Function |
|---|---|---|
| 1 | D/T | I2C data line SDA<br>UART-TX |
| 2 | C/R | I2C clock line SCL<br>UART-RX |

| Num | Label | Function |
|---|---|---|
| 3 | - | Ground |
| 4 | + | VCC |
| 5 | PPS | GNSS Chip pulse output per second |
| 6 | INT | Active-low interrupt/1Hz square wave output |
| 7 | 32K | 32.768KHz pulse output |

# Arduino Tutorial

## Requirements

- **Hardware**
  - Gravity: GNSS Position & Timing Module (https://www.dfrobot.com/product-2815.html) *1
  - DFRduino UNO R3 (https://www.dfrobot.com.cn/goods-521.html) *1
- **Software**
  - Arduino IDE
  - Download and install DFRobot_GNSS and RTC library (https://github.com/cdjq/DFRobot_GNSSAndRTC)
- **Connection Diagram**




## Sample code 1 - Get GNSS Position

Get parameters such as latitude, longitude, number of satellites, number of satellites, etc.
This sample code for I2C communication.

```
/*!
 * @file  getGNSS.ino
 * @brief Get gnss simple data
 * @copyright Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @license The MIT License (MIT)
 * @author [qsjhyy](yihuan.huang@dfrobot.com)
 * @version V1.0
 * @date 2022-08-30
 * @url https://github.com/DFRobot/DFRobot_GNSSAndRTC
 */

#include "DFRobot_GNSSAndRTC.h"

#define I2C_COMMUNICATION  //use I2C for communication, but use the serial port for communica

#ifdef  I2C_COMMUNICATION
DFRobot_GNSSAndRTC_I2C gnss(&Wire, MODULE_I2C_ADDRESS);
#else
 /* ---------------------------------------------------------------------------------------
  *    board   |             MCU                | Leonardo/Mega2560/M0 |    UNO    | ESP8266
  *     VCC    |            3.3V/5V             |        VCC           |    VCC    |   VCC
  *     GND    |              GND               |        GND           |    GND    |   GND
  *     RX     |              TX                |     Serial1 TX1      |     5     |   5/D6
  *     TX     |              RX                |     Serial1 RX1      |     4     |   4/D7
  * ---------------------------------------------------------------------------------------
  /* Baud rate cannot be changed */
#if defined(ARDUINO_AVR_UNO) || defined(ESP8266)
SoftwareSerial mySerial(4, 5);
DFRobot_GNSSAndRTC_UART gnss(&mySerial, UART_BAUDRATE);
#elif defined(ESP32)
DFRobot_GNSSAndRTC_UART gnss(&Serial1, UART_BAUDRATE,/*rx*/D2,/*tx*/D3);
#else
DFRobot_GNSSAndRTC_UART gnss(&Serial1, UART_BAUDRATE);
#endif
#endif
```

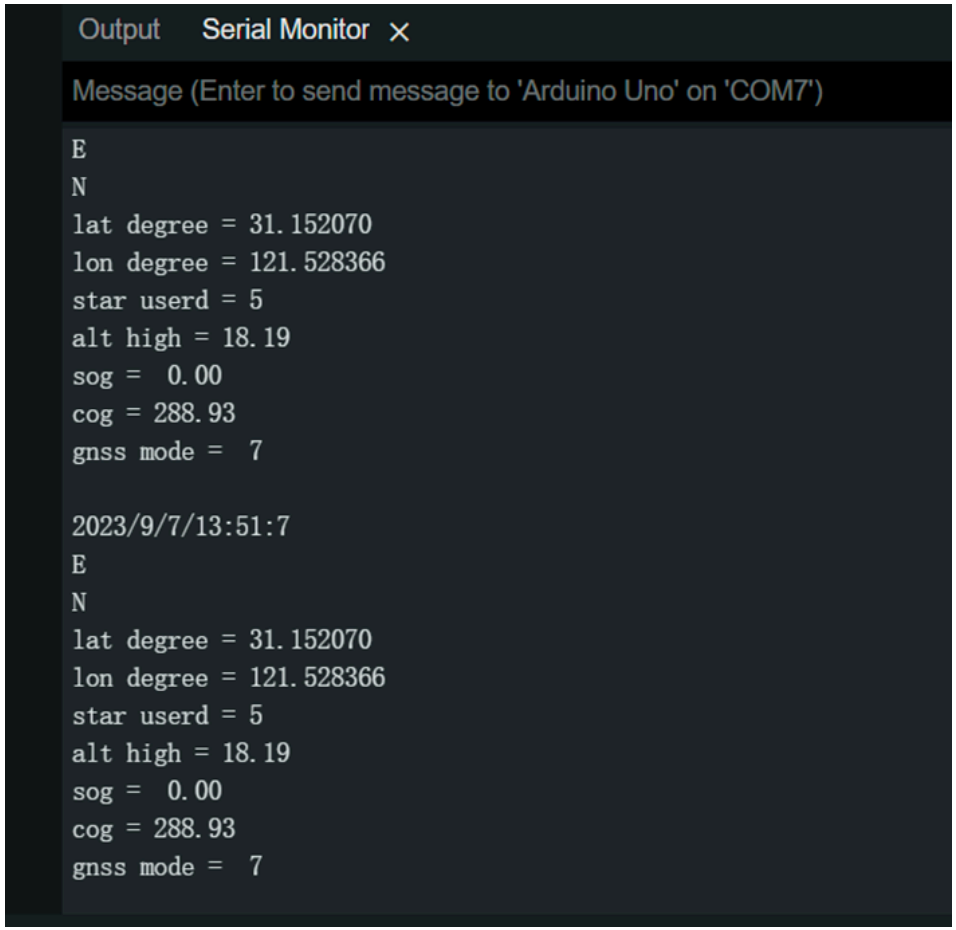
```
void setup()
{
    Serial.begin(115200);
    while (!gnss.begin()) {
        Serial.println("NO Deivces !");
        delay(1000);
}

    gnss.enablePower();      // Enable gnss power

    /** Set GNSS to be used
     *   eGPS              use gps
     *   eBeiDou           use beidou
     *   eGPS_BeiDou       use gps + beidou
     *   eGLONASS          use glonass
     *   eGPS_GLONASS      use gps + glonass
     *   eBeiDou_GLONASS   use beidou +glonass
     *   eGPS_BeiDou_GLONASS use gps + beidou + glonass
     */
    gnss.setGnss(gnss.eGPS_BeiDou_GLONASS);

    // gnss.disablePower();      // Disable GNSS, the data will not be refreshed after disabl
}

void loop()
{
    DFRobot_GNSSAndRTC::sTim_t utc = gnss.getUTC();
    DFRobot_GNSSAndRTC::sTim_t date = gnss.getDate();
    DFRobot_GNSSAndRTC::sLonLat_t lat = gnss.getLat();
    DFRobot_GNSSAndRTC::sLonLat_t lon = gnss.getLon();
    double high = gnss.getAlt();
    uint8_t starUserd = gnss.getNumSatUsed();
    double sog = gnss.getSog();
    double cog = gnss.getCog();
```

```
    Serial.println("");
    Serial.print(date.year);
    Serial.print("/");
    Serial.print(date.month);
    Serial.print("/");
    Serial.print(date.date);
    Serial.print("/");
    Serial.print(utc.hour);
    Serial.print(":");
    Serial.print(utc.minute);
    Serial.print(":");
    Serial.print(utc.second);
    Serial.println();
    Serial.println((char)lat.latDirection);
    Serial.println((char)lon.lonDirection);

    // Serial.print("lat DDMM.MMMMM = ");
    // Serial.println(lat.latitude, 5);
    // Serial.print("lon DDDMM.MMMMM = ");
    // Serial.println(lon.lonitude, 5);
    Serial.print("lat degree = ");
    Serial.println(lat.latitudeDegree, 6);
    Serial.print("lon degree = ");
    Serial.println(lon.lonitudeDegree, 6);

    Serial.print("star userd = ");
    Serial.println(starUserd);
    Serial.print("alt high = ");
    Serial.println(high);
    Serial.print("sog =  ");
    Serial.println(sog);
    Serial.print("cog = ");
    Serial.println(cog);
    Serial.print("gnss mode =  ");
    Serial.println(gnss.getGnssMode());
```

```
    delay(1000);
}
```



## Result



# Sample code 2 - Get RTC Time

Getting the time of the on-board RTC chip.
This sample code for I2C communication.

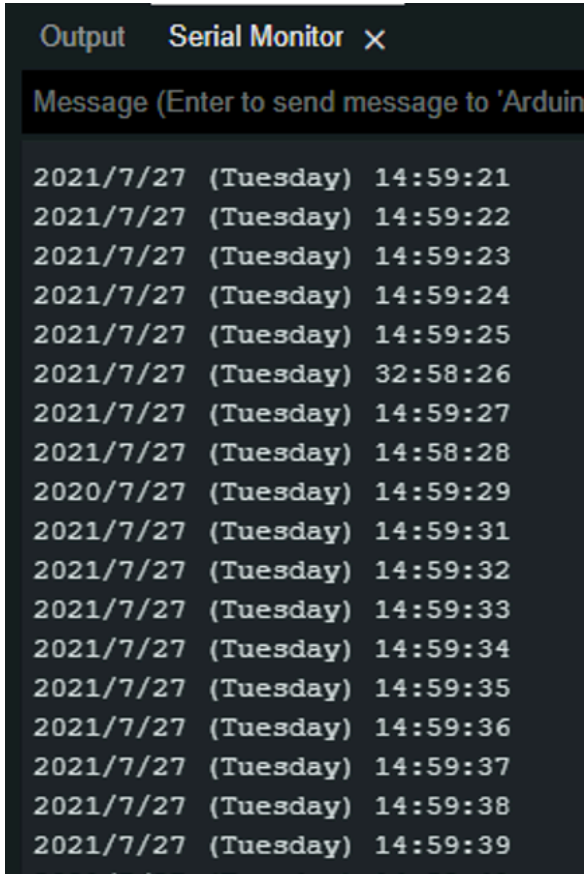
```
/*!
 * @file getTime.ino
 * @brief Run this routine, set internal clock first, and then circularly get clock, temperat
 * @copyright    Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @license      The MIT License (MIT)
 * @author [qsjhyy](yihuan.huang@dfrobot.com)
 * @version V1.0
 * @date 2022-08-30
 * @url https://github.com/DFRobot/DFRobot_GNSSAndRTC
 */
#include "DFRobot_GNSSAndRTC.h"

#define I2C_COMMUNICATION  //use I2C for communication, but use the serial port for communica

#ifdef  I2C_COMMUNICATION
DFRobot_GNSSAndRTC_I2C rtc(&Wire, MODULE_I2C_ADDRESS);
#else
/* ---------------------------------------------------------------------------------------
      *    board   |             MCU                | Leonardo/Mega2560/M0 |    UNO    | ESP8
      *     VCC    |            3.3V/5V             |        VCC           |    VCC    |   VC
      *     GND    |              GND               |        GND           |    GND    |   GN
      *     RX     |              TX                |     Serial1 TX1      |     5     |   5/
      *     TX     |              RX                |     Serial1 RX1      |     4     |   4/
      * ---------------------------------------------------------------------------------------
/* Baud rate cannot be changed  */
#if defined(ARDUINO_AVR_UNO) || defined(ESP8266)
SoftwareSerial mySerial(4, 5);
DFRobot_GNSSAndRTC_UART rtc(&mySerial, UART_BAUDRATE);
#elif defined(ESP32)
DFRobot_GNSSAndRTC_UART rtc(&Serial1, UART_BAUDRATE,/*rx*/D2,/*tx*/D3);
#else
DFRobot_GNSSAndRTC_UART rtc(&Serial1, UART_BAUDRATE);
#endif
#endif
```

```
void setup()
{
    Serial.begin(115200);
    /*Wait for the chip to be initialized completely, and then exit*/
    while(!rtc.begin()){
        Serial.println("Failed to init chip, please check if the chip connection is fine. ");
        delay(1000);
    }
    rtc.setHourSystem(rtc.e24hours);//Set display format
    rtc.setTime(2021,7,27,14,59,0);//Initialize time
    // //Get internal temperature
    // Serial.print(rtc.getTemperatureC());
    // Serial.println(" C");
    // //Get battery voltage
    // Serial.print(rtc.getVoltage());
    // Serial.println(" V");
}

void loop()
{
    DFRobot_GNSSAndRTC::sTimeData_t sTime;
    sTime = rtc.getRTCTime();
    Serial.print(sTime.year, DEC);//year
    Serial.print('/');
    Serial.print(sTime.month, DEC);//month
    Serial.print('/');
    Serial.print(sTime.day, DEC);//day
    Serial.print(" (");
    Serial.print(sTime.week);//week
    Serial.print(") ");
    Serial.print(sTime.hour, DEC);//hour
    Serial.print(':');
    Serial.print(sTime.minute, DEC);//minute
    Serial.print(':');
    Serial.print(sTime.second, DEC);//second
    Serial.println(' ');
```

```
    /*Enable 12-hour time format*/
    // Serial.print(rtc.getAMorPM());
    // Serial.println();
    delay(1000);
}
```



## 结果



# Sample code 3 - Calibrating RTC time using GNSS

Calibrate the RTC chip by GNSS time. The calibration process takes 2 to 3 seconds, during which time the time of this module cannot be acquired properly.
Serial monitor prints ""Calibration success"" when calibration is complete.
This sample code is for I2C communication

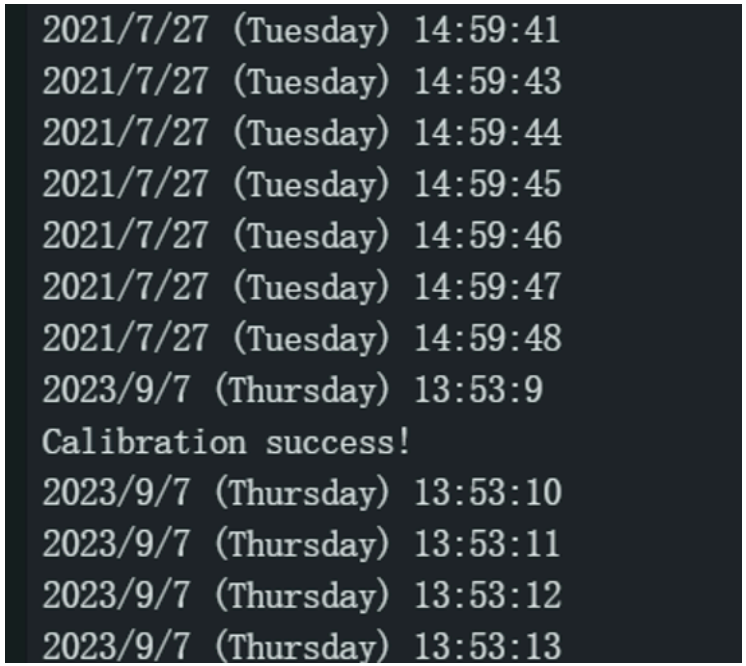
```
/*!
 * @file gnssCalibRTC.ino
 * @brief Run this routine, calibration internal clock first, and then circularly get clock
 * @copyright    Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @license      The MIT License (MIT)
 * @author [qsjhyy](yihuan.huang@dfrobot.com)
 * @version V1.0
 * @date 2022-08-30
 * @url https://github.com/DFRobot/DFRobot_GNSSAndRTC
 */
#include "DFRobot_GNSSAndRTC.h"

#define I2C_COMMUNICATION  //use I2C for communication, but use the serial port for communica

#ifdef  I2C_COMMUNICATION
DFRobot_GNSSAndRTC_I2C rtc(&Wire, MODULE_I2C_ADDRESS);
#else
/* ---------------------------------------------------------------------------------------
      *    board   |             MCU                | Leonardo/Mega2560/M0 |    UNO    | ESP8
      *     VCC    |            3.3V/5V             |        VCC           |    VCC    |   VC
      *     GND    |              GND               |        GND           |    GND    |   GN
      *     RX     |              TX                |     Serial1 TX1      |     5     |   5/
      *     TX     |              RX                |     Serial1 RX1      |     4     |   4/
* ---------------------------------------------------------------------------------------
/* Baud rate cannot be changed  */
#if defined(ARDUINO_AVR_UNO) || defined(ESP8266)
SoftwareSerial mySerial(4, 5);
DFRobot_GNSSAndRTC_UART rtc(&mySerial, UART_BAUDRATE);
#elif defined(ESP32)
DFRobot_GNSSAndRTC_UART rtc(&Serial1, UART_BAUDRATE,/*rx*/D2,/*tx*/D3);
#else
DFRobot_GNSSAndRTC_UART rtc(&Serial1, UART_BAUDRATE);
#endif
#endif
```

```
void setup()
{
    Serial.begin(115200);
    /*Wait for the chip to be initialized completely, and then exit*/
    while (!rtc.begin()) {
        Serial.println("Failed to init chip, please check if the chip connection is fine. ");
        delay(1000);
    }
    rtc.setHourSystem(rtc.e24hours);//Set display format

    /**
     * @brief Calibrate RTC immediately with GNSS
     * @note This is a single calibration;
     * @n If the GNSS module signal is weak, time calibration may encounter issues.
     * @return None
     */
    // rtc.calibRTC();

    /**
    * @brief The loop automatically performs GNSS timing based on the set interval
    * @param hour Automatic calibration of the time interval. range: 0~255, unit: hour.
    * @note When set to zero, automatic time calibration is disabled.
    * @n Enabling it will trigger an immediate calibration.
    * @n If the GNSS module signal is weak, time calibration may encounter issues.
    * @return None
    */
    rtc.calibRTC(1);
}
    uint8_t underCalibCount = 0;

void loop()
{
    /**
    * @brief Current clock calibration status
    * @param mode By default, it is set to true, indicating access to the calibration status
    * @n  If continuous calibration for one minute does not return a successful calibration,
```

```
* @n  you can pass in false to manually terminate this calibration session.
* @return uint8_t type, indicates current clock calibration status
* @retval 0 Not calibrated
* @retval 1 Calibration complete
* @retval 2 Under calibration
* @note Note: To avoid affecting subsequent calibration status,
* @n    "Calibration completed Status (1)" is automatically zeroed after a successful rea
*/
uint8_t status = rtc.calibStatus();
if (DFRobot_GNSSAndRTC::eCalibComplete == status) {
    underCalibCount = 0;
    Serial.println("Calibration success!");
} else if (DFRobot_GNSSAndRTC::eUnderCalib == status) {
    underCalibCount += 1;
    if (60 <= underCalibCount) {   // If the calibration fails for a long time, manually
        rtc.calibStatus(false);
        underCalibCount = 0;
        Serial.println("Calibration failed!");
        Serial.println("It may be due to weak satellite signals.");
        Serial.println("Please proceed to an open outdoor area for time synchronization."
    }
}
DFRobot_GNSSAndRTC::sTimeData_t sTime;
sTime = rtc.getRTCTime();
Serial.print(sTime.year, DEC);//year
Serial.print('/');
Serial.print(sTime.month, DEC);//month
Serial.print('/');
Serial.print(sTime.day, DEC);//day
Serial.print(" (");
Serial.print(sTime.week);//week
Serial.print(") ");
Serial.print(sTime.hour, DEC);//hour
Serial.print(':');
Serial.print(sTime.minute, DEC);//minute
Serial.print(':');
```

```
    Serial.print(sTime.second, DEC);//second
    Serial.println(' ');
    /*Enable 12-hour time format*/
    // Serial.print(rtc.getAMorPM());
    // Serial.println();

    // In addition to data acquisition and other time consuming, the delay of 900ms makes eac
    delay(900);
}
```

## Result



## More API function list

```
/**
 * @fn calibRTC(void)
 * @brief Calibrate RTC immediately with GNSS
 * @note This is a single calibration;
 * @n If the GNSS module signal is weak, time calibration may encounter issues.
 * @return None
 */
   void calibRTC(void);

/**
 * @fn calibRTC(uint8_t hour)
 * @brief The loop automatically performs GNSS timing based on the set interval
 * @param hour Automatic calibration of the time interval. range: 0~255, unit: hour.
 * @note When set to zero, automatic time calibration is disabled.
 * @n Enabling it will trigger an immediate calibration.
 * @n If the GNSS module signal is weak, time calibration may encounter issues.
 * @return None
 */
   void calibRTC(uint8_t hour);

/**
 * @fn calibStatus
 * @brief Current clock calibration status
 * @param mode By default, it is set to true, indicating access to the calibration status o
 * @n  If continuous calibration for one minute does not return a successful calibration,
 * @n  you can pass in false to manually terminate this calibration session.
 * @return uint8_t type, indicates current clock calibration status
 * @retval 0 Not calibrated
 * @retval 1 Calibration complete
```

```
 * @retval 2 Under calibration
 * @note Note: To avoid affecting subsequent calibration status,
 * @n    "Calibration completed Status (1)" is automatically zeroed after a successful read
 */
  uint8_t calibStatus(bool mode = true);

/**
 * @fn setAlarm
 * @brief set an Alarm
 * @param year 2000~2099
 * @param month 1~12
 * @param day 1~31
 * @return None
 */
  void setAlarm(uint16_t year, uint8_t month, uint8_t day);

/**
 * @brief clearAlarm
 */
  void clearAlarm(void);

/**
 * @fn enable32k
 * @brief enable the 32.768kHz output of the Mdule
 * @return None
 */
 void enable32k();


/**
 * @fn disable32k
 * @brief disable the 32.768kHz output of the Mdule
 * @return None
 */
 void disable32k();
```

```
/**
 * @fn countDown
 * @brief CountDown
 * @param second  countdown time 0-0xffffff
 */
 void countDown(uint32_t second);

/**
 * @fn getUTC
 * @brief Get utc standard time
 * @return sTim_t type, represents the returned hour, minute and second
 * @retval sTim_t.hour hour
 * @retval sTim_t.minute minute
 * @retval sTim_t.second second
 */
  sTim_t getUTC(void);


/**
 * @fn enablePower
 * @brief Enable gnss power
 * @return null
 */
 void enablePower(void);

/**
 * @fn disablePower
 * @brief Disable gnss power
 * @return null
 */
 void disablePower(void);
```

# FAQ

For any questions, advice or cool ideas to share, please visit the DFRobot Forum (https://www.dfrobot.com/forum/).

# More Documents

- RTC datasheet (Chinese) (https://img.dfrobot.com.cn/wiki/62b2fb5caa613609f271523c/fe9f748e802db47b4dceaf9300f9c480.pdf)
- GNSS datasheet (Chinese) (https://img.dfrobot.com.cn/wiki/62b2fb5caa613609f271523c/056d1e004b84985e71d94723324c1ff8.pdf)
- Schematic (https://img.dfrobot.com.cn/wiki/62b2fb5caa613609f271523c/349fe1581429b4d94a30ec2bb6ca2525.pdf)
- Dimension & Layout (https://img.dfrobot.com.cn/wiki/62b2fb5caa613609f271523c/fbf22d786210f181f639a2ab818fe64e.jpg)

Get Gravity: GNSS Positioning & Timing Module (https://www.dfrobot.com.cn/goods-3799.html)