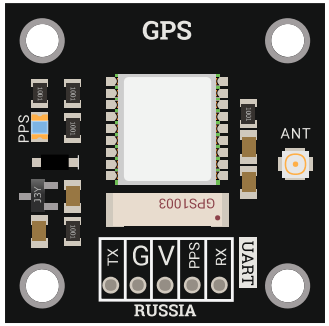
Работа с GPS модулем



Общие сведения:

Trema GPS модуль ATGM336H - является навигационным устройством позволяющим определить свои координаты по широте, долготе и высоте. Дополнительно модуль способен определить текущую дату, время, скорость и направление передвижения.

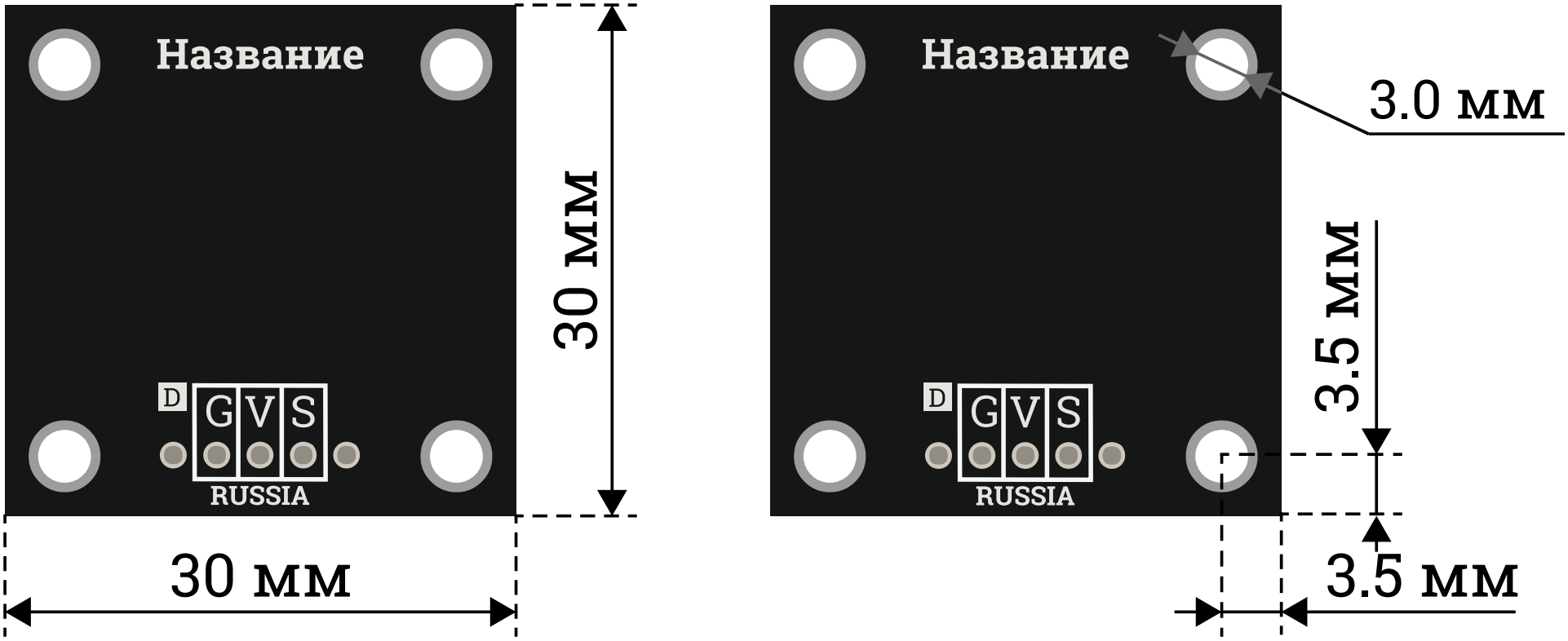
Модуль получает данные на основе информации поступающей со спутников навигационных систем GPS (США), Глонасс (Россия) и Beidou (Китай).

Модуль самостоятельно обрабатывает полученную информацию и передает данные по шине UART в виде текстовых сообщений в формате протокола NMEA 0183, отличается низким энергопотреблением и высокой чувствительностью.

Спецификация:

- Напряжение питания: 3,3 В или 5 В, поддерживаются оба напряжения.
- Питание резервной батареи: 3 В.
- Ток потребляемый модулем: до 25 мА.
- Интерфейс: UART.
- Скорость шины UART: 4800, 9600 (по умолчанию), 19200, 38400, 57600, 115200 бит/с.
- Конфигурация шины UART: 8 бит данных, без проверки четности, с одним стоповым битом.
- Уровень логической 1 на линиях шины UART: 3,3 В.
- Протокол передачи данных: NMEA 0183.
- Частота обновления выводимых данных: от 1 (по умолчанию) до 10 Гц.
- Поддерживаемые навигационные системы: GPS (США), Глонасс (Россия) и Beidou (Китай).
- Время холодного старта: ≤ 35 сек.
- Время горячего старта: ≤ 1 сек.
- Точность позиционирования: < 2 м.
- Точность измерения скорости: < 0,1 м/с.

- Максимальная высота: 18000 м.
- Рабочая температура: от -40 до +85 °C.
- Габариты: 30 x 30 мм.
- Вес: 15 г.



Подключение:

На плате модуля расположен разъем из 5 выводов.

- **TX** - выход данных шины UART от модуля. Подключается к выводу RX Arduino.
- **RX** - вход данных шины UART в модуль. Подключается к выводу TX Arduino.

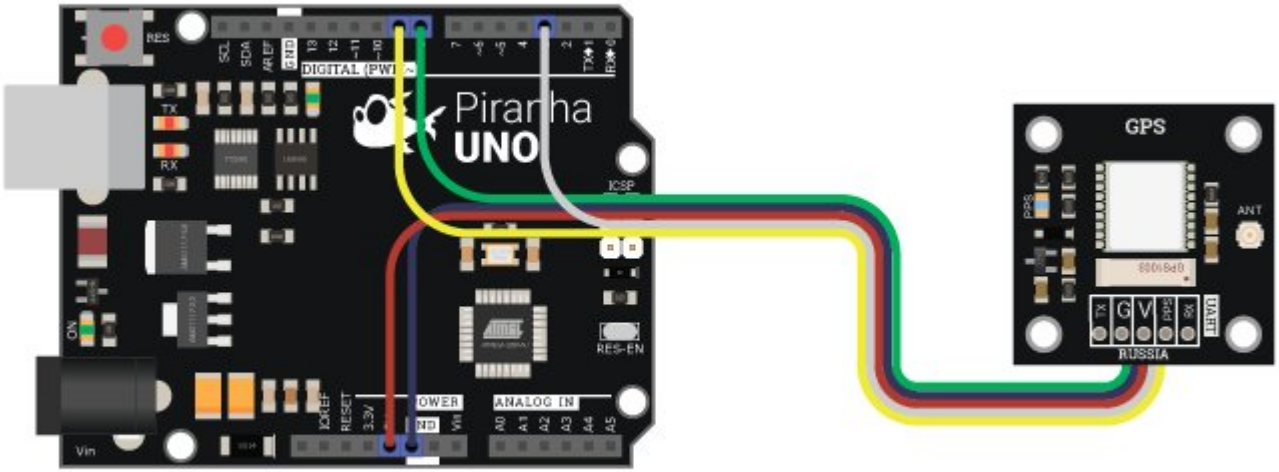
- **Vcc** - вход питания 3,3 или 5 В.
- **GND** - общий вывод питания.
- **PPS** - выход меандра с частотой 1 Гц. Передний фронт импульсов совпадает с временем UTC.

При подключении модуля не к аппаратной, а к программной шине UART, вы сами назначаете выводы TX и RX Arduino к которым подключается модуль.

Модуль удобно подключать 5 способами, в зависимости от ситуации:

Способ - 1: Используя провода, Piranha UNO и программный UART

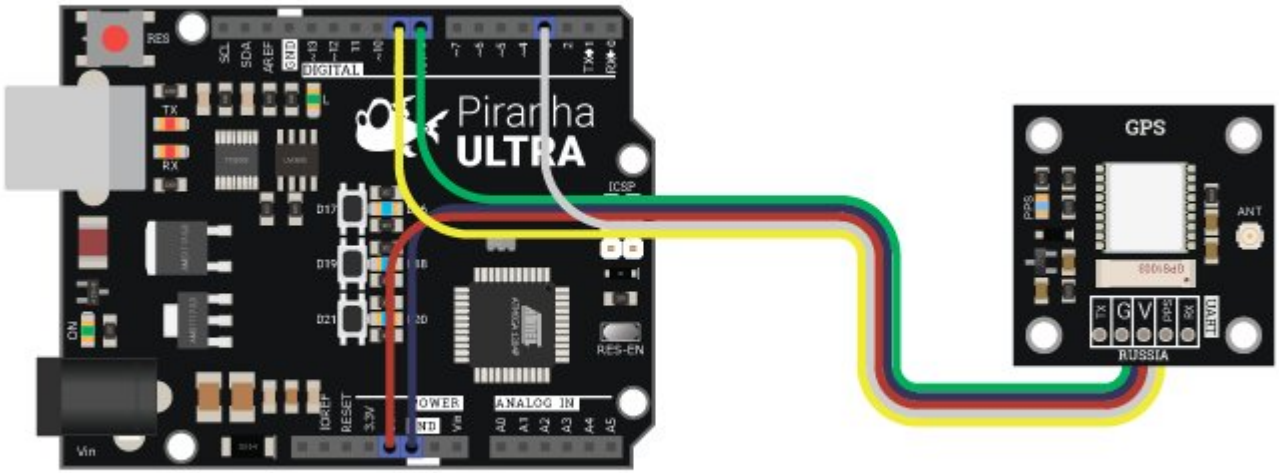
Используя провода «Папа — Мама», подключаем напрямую к контроллеру Piranha UNO.



Способ - 2: Используя провода, Piranha ULTRA и аппаратный UART

Используя провода «Папа — Мама», подключаем напрямую к контроллеру Piranha ULTRA.

С данным подключением будет использоваться второй аппаратный UART на Piranha ULTRA. Стоит заметить, что программный порт на UNO безошибочно работает на скорости до 57600 бод, в то время как аппаратный без проблем может работать на скорости 115200, вдвое большей.

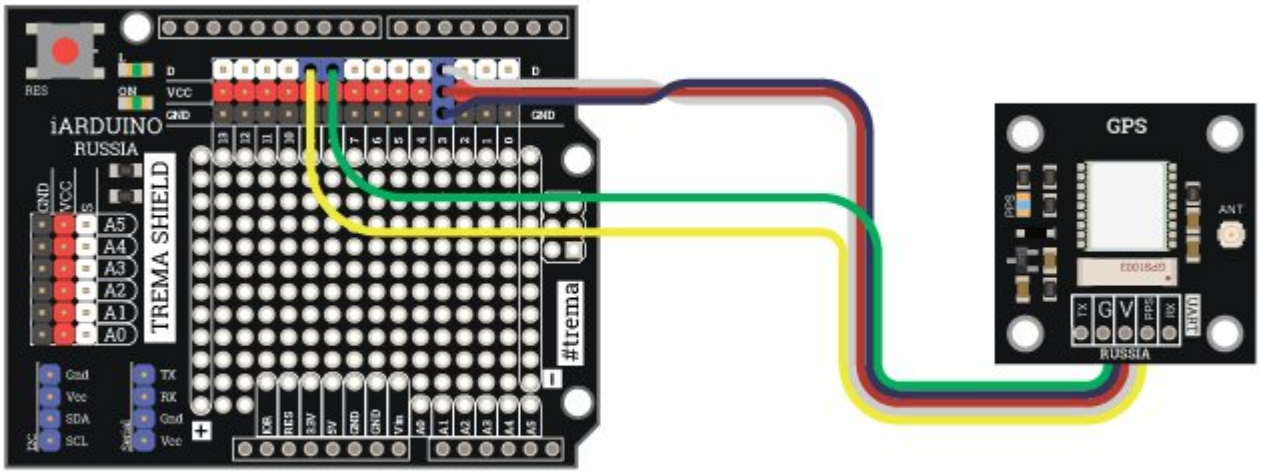


Способ - 3: Используя Trema Set Shield

При таком подключении можно использовать программный UART на 8 и 9 цифровых выводах. Так же на этих выводах находиться второй аппаратный последовательный порт Piranha ULTRA, что ещё больше упрощает работу с модулем.

Способ - 4: Используя проводной шлейф и Shield

Используя 2-х и 3-х проводные шлейфы, к Trema Shield, Trema-Power Shield, Motor Shield, Trema Shield NANO и тд.



Способ - 5: Напрямую к ПК через USB-UART

Питание:

Входное напряжение питания модуля 3,3В или 5В постоянного тока (поддерживаются оба напряжения питания), подаётся на выводы Vcc и GND.

Подробнее о модуле:

Модуль построен на базе чипа AT6558 включённого в сборку AT6558, снабжён антенной GPS1003, разъемом IPX позволяющим подключать внешнюю антенну, разъемом для подключения батарейки резервного питания и снабжён собственным стабилизатором напряжения. Модуль отличается низким энергопотреблением и высокой чувствительностью. Модуль получает информацию от спутников навигационных систем GPS, Глонасс, Beidou, обрабатывает её и передает выходные данные сформированные в формате протокола NMEA 0183 по шине UART в виде текстовых сообщений.

Модуль позволяет:

- Получать широту, долготу и высоту над уровнем моря.
- Получать скорость и курс на истинный полюс.
- Получать дату и время UTC.
- Получать количество наблюдаемых спутников и спутников участвующих в позиционировании.
- Получать данные о спутниках: уровень приёма, положение и тип навигационной системы.

- Получать геометрические факторы ухудшения точности и точность позиционирования.
- Определять ошибку определения координат, даты, времени, скорости и курса.
- Настраивать скорость передачи данных по шине UART.
- Настраивать частоту обновления выводимых данных.
- Выбирать версию протокола NMEA 0183 для формирования отправляемых сообщений.
- Настраивать состав сообщений NMEA 0183.
- Выбирать спутниковые навигационные системы, данные которых требуется получать.
- Выбирать динамическую модель навигационной платформы.
- Перезагружать модуль с выбором типа старта (холодный / горячий).
- Перезагружать модуль со сбросом настроек в заводские.
- Сохранять настройки в энергонезависимую память модуля.

Для работы с Trema GPS модулем - ATGM336H, нами разработано две библиотеки:

- Библиотека iarduino_GPS_ATGM336 позволяет настроить работу Trema GPS модуля ATGM336H (задать скорость UART, частоту обновления данных, версию NMEA 0183, настроить состав выводимых сообщений, выбрать спутниковые навигационные системы, указать динамическую модель, перезагрузить модуль или сохранить его настройки).
- Библиотека iarduino_GPS_NMEA позволяет получать данные из текстовых сообщений NMEA 0183 отправляемых GPS-модулем по шине UART (получить широту, долготу, высоту, скорость, курс, дату, время, количество спутников и данные о них).

Подробнее про установку библиотеки читайте в нашей инструкции.

Примеры:

В данном разделе раскрыты примеры настройки работы Trema GPS модуля ATGM336H при помощи библиотеки iarduino_GPS_ATGM336 и получения данных отправляемых данным модулем при помощи библиотеки iarduino_GPS_NMEA по аппаратной шине UART-1. Обе библиотеки позволяют работать не только с аппаратной, но и с программной шиной UART.

О том как настроить скетч для работы с программной шиной UART можно узнать ниже, из раздела «Подключение библиотеки».

Примеры работы с программной и аппаратной шиной UART доступны из меню Arduino IDE:

- Файл / Примеры / iarduino GPS NMEA (парсер) - *без настройки модуля*.
- Файл / Примеры / iarduino GPS ATGM336 (навигационный модуль).

Примеры получения данных библиотекой iarduino_GPS_NMEA без настройки модуля, доступны по ссылке Wiki - Парсер протокола NMEA.

Получение координат:

Пример выводит координаты широты и долготы в градусах, 5 раз в секунду.

```
#include <iarduino_GPS_NMEA.h>                  //  Подключаем библиотеку для расшифровки строк протокола NMEA получаемых г
#include <iarduino_GPS_ATGM336.h>               //  Подключаем библиотеку для настройки параметров работы GPS модуля ATGM33
                                                //
iarduino_GPS_NMEA    gps;                       //  Объявляем объект gps         для работы с функциями и методами библиоте
iarduino_GPS_ATGM336 SettingsGPS;               //  Объявляем объект SettingsGPS для работы с функциями и методами библиоте
                                                //
void setup(){                                   //
//   Инициируем работу библиотек:               //
     Serial.begin(9600);                        //  Инициируем работу с аппаратной шиной UART для вывода данных в монитор г
     SettingsGPS.begin(Serial1);                //  Инициируем работу с GPS модулем по указанной шине UART. Функция сама ог
     gps.begin(Serial1);                        //  Инициируем расшифровку строк NMEA указав объект используемой шины UART
//   Настраиваем работу модуля:                 //
     SettingsGPS.baudrate(9600);                //  Устанавливаем скорость передачи данных модулем и скорость работы шины S
     SettingsGPS.system(GPS_GP, GPS_GL);        //  Указываем что данные нужно получать от спутников навигационных систем G
     SettingsGPS.composition(NMEA_RMC);         //  Указываем что каждый пакет данных NMEA должен содержать только одну стр
     SettingsGPS.model(GPS_PORTABLE);           //  Указываем что модуль используется в портативном устройстве.
     SettingsGPS.updaterate(10);                //  Указываем обновлять данные 10 раз в секунду. Функция gps.read() читает
}                                               //
                                                //
void loop(){                                    //
//   Читаем данные:                             //
     gps.read();                                //
//   Проверяем достоверность координат:         //
```

```
    if(gps.errPos){                                 //  Если данные не прочитаны (gps.errPos=1) или координаты недостоверны (gp
      Serial.println("Координаты недостоверны");    //  Выводим сообщение об ошибке.
      delay(2000); return;                          //  Ждём 2 секунды и выполняем функцию loop() с начала.
    }                                               //
//  Выводим текущие координаты:                     //
    Serial.print("Широта: ");                       //
    Serial.print(gps.latitude, 5);                  //  Выводим широту в градусах.
    Serial.print("°, ");                            //
    Serial.print("Долгота: ");                      //
    Serial.print(gps.longitude, 5);                 //  Выводим долготу в градусах.
    Serial.println("°.");                           //
}                                                   //
```

Обратите внимание на то, что в примере не указана скорость шины `Serial1`, так как она автоматически настраивается на скорость модуля функцией `SettingsGPS.begin(Serial1);`, а далее устанавливается функцией `SettingsGPS.baudrate(9600);`.

Состав сообщения NMEA 0183 отправляемого модулем, урезан обращением к функции `SettingsGPS.composition();` до одной строки с идентификатором `NMEA_RMC`, вместо данного идентификатора можно указать `NMEA_GGA` или `NMEA_GLL`, так как в их строках так же содержится информация о широте и долготе. Уменьшение строк в составе сообщения NMEA приводит к уменьшению количества данных содержащихся в нём. Например, строка с идентификатором `NMEA_RMC` не содержит значение высоты над уровнем моря и данные о спутниках.

Получение скорости и курса:

Пример выводит скорость и курс, 5 раз в секунду.

```
#include <iarduino_GPS_NMEA.h>                  //  Подключаем библиотеку для расшифровки строк протокола NMEA получаемых п
#include <iarduino_GPS_ATGM336.h>               //  Подключаем библиотеку для настройки параметров работы GPS модуля ATGM33
                                                //
iarduino_GPS_NMEA    gps;                       //  Объявляем объект gps         для работы с функциями и методами библиоте
iarduino_GPS_ATGM336 SettingsGPS;               //  Объявляем объект SettingsGPS для работы с функциями и методами библиоте
```

```
                                                    //
void setup(){                                       //
//   Инициируем работу библиотек:                   //
     Serial.begin(9600);                            //  Инициируем работу с аппаратной шиной UART для вывода данных в монитор п
     SettingsGPS.begin(Serial1);                    //  Инициируем работу с GPS модулем по указанной шине UART. Функция сама оп
     gps.begin(Serial1);                            //  Инициируем расшифровку строк NMEA указав объект используемой шины UART
//   Настраиваем работу модуля:                     //
     SettingsGPS.baudrate(9600);                    //  Устанавливаем скорость передачи данных модулем и скорость работы шины S
     SettingsGPS.system(GPS_GP, GPS_GL);            //  Указываем что данные нужно получать от спутников навигационных систем G
     SettingsGPS.composition(NMEA_VTG);             //  Указываем что каждый пакет данных NMEA должен содержать только одну стр
     SettingsGPS.model(GPS_VEHICLE);                //  Указываем что модуль используется в автомобиле.
     SettingsGPS.updaterate(10);                    //  Указываем обновлять данные 10 раз в секунду. Функция gps.read() читает
}                                                   //
                                                    //
void loop(){                                        //
//   Читаем данные:                                 //
     gps.read();                                    //
//   Проверяем достоверность скорости и курса:      //
     if(gps.errCrs){                                //  Если данные не прочитаны (gps.errCrs=1) или скорость и курс недостоверн
       Serial.println("Скорость недостоверна");     //  Выводим сообщение об ошибке.
       delay(2000); return;                         //  Ждём 2 секунды и выполняем функцию loop() с начала.
     }                                              //
//   Выводим текущие скорость и курс:               //
     Serial.print("Скорость: ");                    //
     Serial.print(gps.speed);                       //  Выводим скорость в км/ч.
     Serial.print("км/ч. ");                        //
     Serial.print("Курс: ");                        //
     Serial.print(gps.course);                      //  Выводим курс в градусах.
     Serial.println("°.\r\n");                      //
}                                                   //
```

Обратите внимание на то, что в примере не указана скорость шины `Serial1`, так как она автоматически настраивается на скорость модуля функцией `SettingsGPS.begin(Serial1);`, а далее устанавливается функцией `SettingsGPS.baudrate(9600);`.

Состав сообщения NMEA 0183 отправляемого модулем, урезан обращением к функции `SettingsGPS.composition();` до одной строки с идентификатором `NMEA_VTG`, вместо данного идентификатора можно указать `NMEA_RMC`, так как в этой строке так же содержится информация о скорости и курсе. Уменьшение строк в составе сообщения NMEA приводит к уменьшению количества данных содержащихся в нём. Например, строка с идентификатором `NMEA_VTG` содержит информацию только о скорости и курсе, без координат, даты, времени и т.д.

Получение ссылки с координатами на карте:

Пример выводит ссылку на Yandex карту 1 раз в секунду.

```
#include <iarduino_GPS_NMEA.h>                  //  Подключаем библиотеку для расшифровки строк протокола NMEA получаемых п
#include <iarduino_GPS_ATGM336.h>               //  Подключаем библиотеку для настройки параметров работы GPS модуля ATGM33
                                                //
iarduino_GPS_NMEA    gps;                       //  Объявляем объект gps         для работы с функциями и методами библиоте
iarduino_GPS_ATGM336 SettingsGPS;               //  Объявляем объект SettingsGPS для работы с функциями и методами библиоте
                                                //
void setup(){                                   //
//   Инициируем работу библиотек:               //
     Serial.begin(9600);                        //  Инициируем работу с аппаратной шиной UART для вывода данных в монитор п
     SettingsGPS.begin(Serial1);                //  Инициируем работу с GPS модулем по указанной шине UART. Функция сама оп
     gps.begin(Serial1);                        //  Инициируем расшифровку строк NMEA указав объект используемой шины UART
//   Настраиваем работу модуля:                 //
     SettingsGPS.baudrate(9600);                //  Устанавливаем скорость передачи данных модулем и скорость работы шины S
     SettingsGPS.system(GPS_GP, GPS_GL);        //  Указываем что данные нужно получать от спутников навигационных систем G
     SettingsGPS.composition(NMEA_RMC);         //  Указываем что каждый пакет данных NMEA должен содержать только одну стр
     SettingsGPS.model(GPS_VEHICLE);            //  Указываем что модуль используется в автомобиле.
     SettingsGPS.updaterate(2);                 //  Указываем обновлять данные 2 раза в секунду. Функция gps.read() читает
}                                               //
                                                //
```

```
void loop(){                                    //
//   Читаем данные:                             //
     gps.read();                                //
//   Проверяем достоверность координат:         //
     if(gps.errPos){                            //  Если данные не прочитаны (gps.errPos=1) или координаты недостоверны (gp
       Serial.println("Координаты недостоверны"); //  Выводим сообщение об ошибке.
       delay(2000); return;                     //  Ждём 2 секунды и выполняем функцию loop() с начала.
     }                                          //
//   Выводим ссылку на Yandex карты:            //
     Serial.print("http://maps.yandex.ru/");    //
//   Координаты центра экрана:                  //
     Serial.print("?ll=");                    //
     Serial.print(gps.longitude,5);             //  Долгота.
     Serial.print(",");                         //  ,
     Serial.print(gps.latitude,5);              //  Широта.
//   Координаты точки на карте:                 //
     Serial.print("&pt=");                    //
     Serial.print(gps.longitude,5);             //  Долгота.
     Serial.print(",");                         //  ,
     Serial.print(gps.latitude,5);              //  Широта.
//   Тип карты:                                 //
     Serial.print("&l=");                     //
     Serial.print("map");                     //  "map"-схема (по умолчанию), "sat"-спутник, "skl"-гибрид.
//   Масштаб карты:                             //
     Serial.print("&z=");                     //
     Serial.print("18");                      //  от 2-мир, до 19-дом, по умолчанию 10-город.
//   Завершаем строку ссылки:                   //
     Serial.print("\r\n");                    //
}                                               //
```

Пример выводит ссылку на Google карту 1 раз в секунду.

```ino
#include <iarduino_GPS_NMEA.h>                      //  Подключаем библиотеку для расшифровки строк протокола NMEA получаемых г
#include <iarduino_GPS_ATGM336.h>                   //  Подключаем библиотеку для настройки параметров работы GPS модуля ATGM33
                                                    //
iarduino_GPS_NMEA    gps;                           //  Объявляем объект gps         для работы с функциями и методами библиоте
iarduino_GPS_ATGM336 SettingsGPS;                   //  Объявляем объект SettingsGPS для работы с функциями и методами библиоте
                                                    //
void setup(){                                       //
//   Инициируем работу библиотек:                   //
     Serial.begin(9600);                            //  Инициируем работу с аппаратной шиной UART для вывода данных в монитор г
     SettingsGPS.begin(Serial1);                    //  Инициируем работу с GPS модулем по указанной шине UART. Функция сама ог
     gps.begin(Serial1);                            //  Инициируем расшифровку строк NMEA указав объект используемой шины UART
//   Настраиваем работу модуля:                     //
     SettingsGPS.baudrate(9600);                    //  Устанавливаем скорость передачи данных модулем и скорость работы шины S
     SettingsGPS.system(GPS_GP, GPS_GL);            //  Указываем что данные нужно получать от спутников навигационных систем G
     SettingsGPS.composition(NMEA_RMC);             //  Указываем что каждый пакет данных NMEA должен содержать только одну стр
     SettingsGPS.model(GPS_VEHICLE);                //  Указываем что модуль используется в автомобиле.
     SettingsGPS.updaterate(2);                     //  Указываем обновлять данные 2 раза в секунду. Функция gps.read() читает
}                                                   //
                                                    //
void loop(){                                        //
//   Читаем данные:                                 //
     gps.read();                                    //
//   Проверяем достоверность координат:             //
     if(gps.errPos){                                //  Если данные не прочитаны (gps.errPos=1) или координаты недостоверны (gp
       Serial.println("Координаты недостоверны");   //  Выводим сообщение об ошибке.
       delay(2000); return;                         //  Ждём 2 секунды и выполняем функцию loop() с начала.
     }                                              //
//   Выводим ссылку на Google карты:                //
     Serial.print("https://www.google.ru/maps/");   //
//   Координаты точки на карте:                     //
     Serial.print("place/");                        //
     Serial.print(gps.latitude,5);                  //  Широта.
```

```
    Serial.print(",");                          //  ,
    Serial.print(gps.longitude,5);              //  Долгота.
//  Язык:                                       //
    Serial.print("?hl=");                       //
    Serial.print("ru");                         //  Русский
//  Завершаем строку ссылки:                    //
    Serial.print("\r\n");                       //
}                                               //
```

Обратите внимание на то, что в примере не указана скорость шины `Serial1`, так как она автоматически настраивается на скорость модуля функцией `SettingsGPS.begin(Serial1);`, а далее устанавливается функцией `SettingsGPS.baudrate(9600);`.

Состав сообщения NMEA 0183 отправляемого модулем, урезан обращением к функции `SettingsGPS.composition();` до одной строки с идентификатором `NMEA_RMC`, вместо данного идентификатора можно указать `NMEA_GGA` или `NMEA_GLL`, так как в их строках так же содержится информация о широте и долготе. Уменьшение строк в составе сообщения NMEA приводит к уменьшению количества данных содержащихся в нём. Например, строка с идентификатором `NMEA_RMC` не содержит значение высоты над уровнем моря и данные о спутниках.

Получение текущей даты и времени:

Пример выводит время, дату, день недели и UnixTime, 5 раз в секунду.

```
#include <iarduino_GPS_NMEA.h>                  //  Подключаем библиотеку для расшифровки строк протокола NMEA получаемых г
#include <iarduino_GPS_ATGM336.h>               //  Подключаем библиотеку для настройки параметров работы GPS модуля ATGM33
                                                //
iarduino_GPS_NMEA    gps;                       //  Объявляем объект gps         для работы с функциями и методами библиоте
iarduino_GPS_ATGM336 SettingsGPS;               //  Объявляем объект SettingsGPS для работы с функциями и методами библиоте
                                                //
char* wd[]={"Вс","Пн","Вт","Ср","Чт","Пт","Сб"};//  Определяем массив строк содержащих по две первых буквы из названий дня
                                                //
void setup(){                                   //
```

```
//   Инициируем работу библиотек:                    //
     Serial.begin(9600);                             //  Инициируем работу с аппаратной шиной UART для вывода данных в монитор г
     SettingsGPS.begin(Serial1);                     //  Инициируем работу с GPS модулем по указанной шине UART. Функция сама ог
     gps.begin(Serial1);                             //  Инициируем расшифровку строк NMEA указав объект используемой шины UART
//   Настраиваем работу модуля:                      //
     SettingsGPS.baudrate(9600);                     //  Устанавливаем скорость передачи данных модулем и скорость работы шины S
     SettingsGPS.system(GPS_GP, GPS_GL);             //  Указываем что данные нужно получать от спутников навигационных систем G
     SettingsGPS.composition(NMEA_ZDA);              //  Указываем что каждый пакет данных NMEA должен содержать только одну стр
     SettingsGPS.model(GPS_STATIC);                  //  Указываем что модуль используется стационарно.
     SettingsGPS.updaterate(10);                     //  Указываем обновлять данные 10 раз в секунду. Функция gps.read() читает
}                                                    //
                                                     //
void loop(){                                         //
//   Читаем данные:                                  //
     gps.read();                                     //
//   Выводим время:                                  //
     Serial.print(gps.Hours  ); Serial.print(":"); //  Выводим час.
     Serial.print(gps.minutes); Serial.print(":"); //  Выводим минуты.
     Serial.print(gps.seconds); Serial.print(" "); //  Выводим секунды.
//   Выводим дату:                                   //
     Serial.print(gps.day    ); Serial.print("."); //  Выводим день месяца.
     Serial.print(gps.month  ); Serial.print("."); //  Выводим месяц.
     Serial.print(gps.year   ); Serial.print("г.");//  Выводим год.
//   Выводим день недели:                            //
     Serial.print(" (");                             //
     Serial.print(wd[gps.weekday]);                  //  Выводим день недели.
     Serial.print("), ");                            //
//   Выводим количество секунд с начала эпохи Unix //
     Serial.print("UnixTime: ");                     //
     Serial.print(gps.Unix);                         //  Выводим время UnixTime.
     Serial.print("с.");                             //
//   Выводим информацию о наличии ошибок:            //
     if( gps.errTim ){                               //
```

```
        Serial.print(" Время недостоверно.");     //  Выводим информацию о недостоверном времени.
    }                                             //
    if( gps.errDat ){                             //
        Serial.print(" Дата недостоверна.");      //  Выводим информацию о недостоверной дате.
    }                                             //
//  Завершаем строку:                             //
    Serial.print("\r\n");                         //
}                                                 //
```

Обратите внимание на то, что в примере не указана скорость шины `Serial1`, так как она автоматически настраивается на скорость модуля функцией `SettingsGPS.begin(Serial1);`, а далее устанавливается функцией `SettingsGPS.baudrate(9600);`.

Состав сообщения NMEA 0183 отправляемого модулем, урезан обращением к функции `SettingsGPS.composition();` до одной строки с идентификатором `NMEA_ZDA`, вместо данного идентификатора можно указать `NMEA_RMC`, так как в этой строке так же содержится информация о дате и времени. А в строках с идентификаторами `NMEA_GGA`, `NMEA_GLL`, `NMEA_GST` и `NMEA_DHV` присутствует только время, без даты. Уменьшение строк в составе сообщения NMEA приводит к уменьшению количества данных содержащихся в нём. Например, строка с идентификатором `NMEA_ZDA` содержит только дату и время.

Получение данных о спутниках:

Пример выводит данные о наблюдаемых спутниках: ID, уровень приёма, положение относительно GPS-модуля, тип навигационной системы и флаг участия спутника в позиционировании, 1 раз в 2 секунды.

```
#include <iarduino_GPS_NMEA.h>                    //  Подключаем библиотеку для расшифровки строк протокола NMEA получаемых г
#include <iarduino_GPS_ATGM336.h>                 //  Подключаем библиотеку для настройки параметров работы GPS модуля ATGM33
                                                  //
iarduino_GPS_NMEA    gps;                         //  Объявляем объект gps         для работы с функциями и методами библиоте
iarduino_GPS_ATGM336 SettingsGPS;                 //  Объявляем объект SettingsGPS для работы с функциями и методами библиоте
                                                  //
uint8_t i[30][7];                                 //  Объявляем массив для получения данных о 30 спутниках в формате: {ID спу
char* sa[]={"NoName ","GPS    ","Глонасс"};       //  Определяем массив строк содержащих названия навигационных систем спутни
```

```
                                                    //
void setup(){                                       //
//   Инициируем работу библиотек:                   //
     Serial.begin(9600);                            //  Инициируем работу с аппаратной шиной UART для вывода данных в монитор г
     SettingsGPS.begin(Serial1);                    //  Инициируем работу с GPS модулем по указанной шине UART. Функция сама ог
     gps.begin(Serial1);                            //  Инициируем расшифровку строк NMEA указав объект используемой шины UART
//   Настраиваем работу модуля:                     //
     SettingsGPS.baudrate(9600);                    //  Устанавливаем скорость передачи данных модулем и скорость работы шины S
     SettingsGPS.system(GPS_GP, GPS_GL);            //  Указываем что данные нужно получать от спутников навигационных систем G
     SettingsGPS.composition(NMEA_GSA, NMEA_GSV);   //  Указываем что каждый пакет данных NMEA должен содержать только строки и
     SettingsGPS.model(GPS_PORTABLE);               //  Указываем что модуль используется в портативном устройстве.
     SettingsGPS.updaterate(1);                     //  Указываем обновлять данные 1 раз в секунду. Функция gps.read() читает д
}                                                   //
                                                    //
void loop(){                                        //
//   Читаем данные:                                 //
     gps.read(i);                                   //  Читаем данные с получением информации о спутниках в массив i (чтение мс
//   Выводим информацию о спутниках:                //
     for(uint8_t j=0; j<30; j++){                   //  Проходим по всем элементам массива «i».
        if( i[j][0] ){                              //  Если у спутника есть ID, то выводим информацию о нём:
            if(j<9) Serial.print(0);                //  Выводим ведущий 0.
            Serial.print(j+1);                      //  Выводим номер строки в мониторе последовательного порта.
            Serial.print(") Спутник ");             //
            Serial.print(sa[i[j][2]]);              //  Выводим название навигационной системы спутника.
            Serial.print(" ID: ");                  //
            if(i[j][0]<10) Serial.print(0);         //  Выводим ведущий 0.
            Serial.print(i[j][0]);                  //  Выводим ID спутника.
            Serial.print(". Уровень: ");            //
            if(i[j][1]<10) Serial.print(0);         //  Выводим ведущий 0.
            Serial.print(i[j][1]);                  //  Выводим отношение сигнал/шум (SNR) в дБ.
            Serial.print("дБ. Возвышение: ");       //
            if(i[j][4]<10) Serial.print(0);         //  Выводим ведущий 0.
            Serial.print(i[j][4]);                  //  Выводим возвышение спутника относительно модуля (0°-горизонт ... 90°-зе
```

```
            Serial.print("°. Азимут: ");           //
            if(i[j][5]<100) Serial.print(0);       //  Выводим ведущий 0.
            if(i[j][5]<10) Serial.print(0);        //  Выводим ведущий 0.
            Serial.print(i[j][5]+i[j][6]);         //  Выводим азимут нахождения спутника относительно модуля (0...360°).
            Serial.print("°. ");                   //
            if(i[j][3]){Serial.print("Участвует в позиционировании.");}
            Serial.print("\r\n");                  //
        }                                          //
    }                                              //
    Serial.println("---------------------------"); //
}                                                  //
```

Обратите внимание на то, что в примере не указана скорость шины `Serial1`, так как она автоматически настраивается на скорость модуля функцией `SettingsGPS.begin(Serial1);`, а далее устанавливается функцией `SettingsGPS.baudrate(9600);`.

Состав сообщения NMEA 0183 отправляемого модулем, урезан обращением к функции `SettingsGPS.composition();` до строк с идентификаторами `NMEA_GSA` и `NMEA_GSV`. Уменьшение строк в составе сообщения NMEA приводит к уменьшению количества данных содержащихся в нём. Например, строки с идентификаторами `NMEA_GSA` и `NMEA_GSV` содержат только информацию о спутниках и геометрические факторы ухудшения точности.

Перезагрузка модуля со сбросом его настроек на заводские:

Пример перезагружает модуль со сбросом его настроек (скорость передачи данных, частота обновления выводимых данных, версия протокола NMEA и его состав, используемые спутниковые навигационные системы и динамическая модель) в значения по умолчанию.

```
#include <iarduino_GPS_ATGM336.h>                  //  Подключаем библиотеку для настройки параметров работы GPS модуля ATGM33
iarduino_GPS_ATGM336 SettingsGPS;                  //  Объявляем объект SettingsGPS для работы с функциями и методами библиоте
                                                   //
void setup(){                                      //
     SettingsGPS.begin(Serial1);                   //  Инициируем работу с GPS модулем по указанной шине UART. Функция сама от
     SettingsGPS.reset(GPS_FACTORY_SET);           //  Перезагружаем модуль со сбросом его настроек.
```

```
}                                                   //
                                                    //
void loop(){                                        //
}                                                   //
```

Описание функций библиотеки iarduino_GPS_ATGM336:

В данном разделе описаны функции библиотеки iarduino_GPS_ATGM336 для настройки работы Trema GPS модуля.

Библиотека iarduino_GPS_ATGM336 может использовать как аппаратную, так и программную реализацию шины UART для работы с Trema GPS-модулем.

Подключение библиотеки:

- Если GPS-модуль подключён по аппаратной шине UART (в примере используется UART-1):

```
#include <iarduino_GPS_ATGM336.h>       //   Подключаем библиотеку для настройки параметров работы GPS модуля ATGM336.
#include <iarduino_GPS_NMEA.h>          // * Подключаем библиотеку для расшифровки строк протокола NMEA получаемых по UART.
iarduino_GPS_ATGM336 SettingsGPS;       //   Создаём объект SettingsGPS для работы с функциями и методами библиотеки iarduino_
iarduino_GPS_NMEA    gps;               // * Создаём объект gps         для работы с функциями и методами библиотеки iarduino_
                                        //
void setup(){                           //
     SettingsGPS.begin(Serial1);        //   Инициируем работу с GPS модулем указав класс аппаратной шины UART по которой подк
     gps.begin(Serial1);                // * Инициируем расшифровку строк NMEA указав объект используемой шины UART.
}                                       //   Функция begin() объекта gps должна быть вызвана после обращения к функции begin()
```

Строки со звездочкой в комментарии можно исключить, если вы не собираетесь читать данные при помощи библиотеки iarduino_GPS_NMEA.

- Если GPS-модуль подключён по программной шине UART (в примере используются выводы D8 и D9 для подключения модуля к Arduino):

```
#include <SoftwareSerial.h>              //   Подключаем библиотеку для работы с программным UART.
#include <iarduino_GPS_ATGM336.h>        //   Подключаем библиотеку для настройки параметров работы GPS модуля ATGM336.
#include <iarduino_GPS_NMEA.h>           // * Подключаем библиотеку для расшифровки строк протокола NMEA получаемых по UART.
SoftwareSerial SerialGPS(8, 9);          //   Создаём объект SerialGPS   для работы с функциями и методами библиотеки SoftwareS
iarduino_GPS_ATGM336 SettingsGPS;        //   Создаём объект SettingsGPS для работы с функциями и методами библиотеки iarduino_
iarduino_GPS_NMEA    gps;                // * Создаём объект gps         для работы с функциями и методами библиотеки iarduino_
                                         //
void setup(){                            //
     SettingsGPS.begin(SerialGPS);       //   Инициируем работу с GPS модулем указав объект программной шины UART по которой по
     gps.begin(SerialGPS);               // * Инициируем расшифровку строк NMEA указав объект используемой шины UART.
}                                        //   Функция begin() объекта gps должна быть вызвана после обращения к функции begin()
```

Строки со звездочкой в комментарии можно исключить, если вы не собираетесь читать данные при помощи библиотеки iarduino_GPS_NMEA.

- Вывод TX Arduino подключается к выводу RX модуля, а вывод RX Arduino подключается к выводу TX модуля.

Поробнее про протокол UART

Функция begin();

- Назначение: Инициализация работы с GPS модулем по шине UART.
- Синтаксис: begin(SERIAL);
- Параметры:
  - SERIAL - объект или класс для работы с шиной UART по которой подключён GPS-модуль.
- Возвращаемое значение: bool - результат инициализации (true или false).
- Примечание:
  - Функция должна быть вызвана до обращения к функции begin() библиотеки iarduino_GPS_NMEA, если таковая используется в скетче.
  - Функция определяет наличие GPS модуля на шине UART, узнаёт скорость передачи данных на которую настроен модуль (это может занять несколько секунд) и указывает шине UART работать на обнаруженной скорости.
  - Если шине UART была задана скорость до обращения к данной функции, то функция сначала проверит соответствует ли указанная

скорость для работы с модулем, а если скорость не подходит, то изменит её на скорость передачи данных модуля.

- Пример:

```
SettingsGPS.begin(Serial1); // Инициируем работу с GPS модулем по аппаратной шине UART-1.
```

Функция baudrate();

- Назначение: Установка скорости передачи данных модуля по шине UART.
- Синтаксис: baudrate([СКОРОСТЬ]);
- Параметры:
  - uint32_t СКОРОСТЬ - целое число может принимать следующие значения:
    - 4800 бит/сек.
    - 9600 бит/сек. - *это значение установлено в модуле по умолчанию.*
    - 19200 бит/сек.
    - 38400 бит/сек.
    - 57600 бит/сек.
    - 115200 бит/сек.
- Возвращаемое значение: uint32_t - результат установки скорости модуля (true / false).
- Примечание:
  - Функция устанавливает указанную скорость и модулю и шине UART.
  - Если функцию вызвать без параметра, то функция обнаружит текущую скорость модуля и установит её для шины UART.

    При этом функция вернет:
    - false (0) - если скорость модуля не определена.
    - true (1) - если скорость модуля совпадает с уже установленной скоростью шины UART.
    - СКОРОСТЬ - обнаруженная скорость модуля установленная для шины UART.
- Пример:

```
SettingsGPS.baudrate(9600); // Установить скорость 9600 бит/сек и модулю, и шине UART.
```

Функция updaterate();

- Назначение: Установка частоты обновления сообщений NMEA выводимых модулем.
- Синтаксис: updaterate(ЧАСТОТА);
- Параметры:
  - uint8_t ЧАСТОТА - целое число от 1 до 10 Гц (от 1 до 10 раз в секунду).
- Возвращаемое значение: Нет.
- Примечание:
  - Функция указывает модулю, как часто надо отправлять сообщения NMEA по шине UART.
  - По умолчанию модуль отправляет сообщения NMEA с частотой 1 Гц (1 раз в секунду).
  - Данные читаются функцией `read()` библиотеки iarduino_GPS_NMEA, в два раза медленнее, чем их отправляет модуль. Функция `read()` не знает какое количество строк включено в сообщение NMEA 0183 отправляемое GPS-модулем, по этому читает данные до получения идентификатора уже прочитанной строки, или до истечения времени указанного функцией `timeOut()` той же библиотеки.
- Пример:

```
SettingsGPS.updaterate(5); // Указываем модулю обновлять данные 5 раз в секунду. Функция gps.read() читает данные в 2 раза мед
```

Функция composition();

- Назначение: Установка состава сообщений NMEA 0183 отправляемых модулем.
- Синтаксис: composition(ИДЕНТИФИКАТОРЫ);
- Параметры:
  - uint8_t ИНДЕНТИФИКАТОРЫ - от 1 до 11 идентификаторов строк сообщения NMEA. Идентификаторы указываются через запятую и могут принимать следующие значения:
    - NMEA_GGA - *данные о последнем зафиксированном местоположении.*
    - NMEA_GLL - *географические координаты.*
    - NMEA_GSA - *активные спутники и геометрические факторы ухудшения точности.*
    - NMEA_GSV - *информация о всех наблюдаемых спутниках.*

        - NMEA_RMC - *рекомендуемый минимум навигационных данных*.
        - NMEA_VTG - *скорость и курс относительно земли*.
        - NMEA_ZDA - *дата и время*.
        - NMEA_ANT - *текстовое сообщение о состоянии антенны*.
        - NMEA_DHV - *скорость движения GNSS приемника*.
        - NMEA_TXT - *текстовое сообщение*.
        - NMEA_GST - *статистика ошибок позиционирования*.
- Возвращаемое значение: Нет.
- Примечание:
    - По умолчанию модуль отправляет сообщения NMEA 0183 состоящие из строк с идентификаторами: GGA, GLL, GSA, GSV, RMC, VTG, ZDA и ANT.
    - Уменьшение строк в составе сообщения NMEA приводит к уменьшению количества данных содержащихся в нём, но способствует ускорению передачи всего сообщения.
- Пример:

```
SettingsGPS.composition(NMEA_RMC, NMEA_GGA, NMEA_GSA, NMEA_GSV); // Указываем модулю отправлять сообщения NMEA состоящие из ст
SettingsGPS.composition(NMEA_RMC); // Указываем модулю отправлять сообщения NMEA состоящие из одной строки содержащей идентифи
```

Функция system();

- Назначение: Выбор используемых спутниковых навигационных систем.
- Синтаксис: system(СИСТЕМЫ);
- Параметры:
    - uint8_t СИСТЕМЫ - от 1 до 3 спутниковых навигационных систем, указываются через запятую и могут принимать следующие значения:
        - GPS_GP - *использовать данные поступающие от спутников GPS*.
        - GPS_BD - *использовать данные поступающие от спутников Beidu*.
        - GPS_GL - *использовать данные поступающие от спутников Glonass*.
- Возвращаемое значение: Нет.
- Примечание:

    - По умолчанию модуль использует спутниковые навигационные системы GPS и Glonass.
- Пример:

```
SettingsGPS.system(GPS_GP, GPS_GL); // Использовать данные поступающие от спутников GPS и Глонасс.
SettingsGPS.system(GPS_GL);         // Использовать данные поступающие только от спутников Глонасс.
```

Функция model();

- Назначение: Выбор динамической модели навигационной платформы.
- Синтаксис: model(МОДЕЛЬ);
- Параметр:
    - uint8_t МОДЕЛЬ - может принимать одно из следующих значений:
        - GPS_PORTABLE - *модуль используется в портативном устройстве*.
        - GPS_STATIC - *модуль используется статично*.
        - GPS_WALKING - *модуль используется пешеходом*.
        - GPS_VEHICLE - *модуль используется в автомобиле*.
        - GPS_VOYAGE - *модуль используется на морском судне*.
        - GPS_AVIATION_1G - *модуль используется на воздушном судне < 1g*.
        - GPS_AVIATION_2G - *модуль используется на воздушном судне < 2g*.
        - GPS_AVIATION_4G - *модуль используется на воздушном судне < 4g*.
- Возвращаемое значение: Нет.
- Примечание:
    - Указание динамической модели использования модуля позволяет чипу более точно вычислять его координаты.
    - По умолчанию модуль использует модель GPS_PORTABLE.
- Пример:

```
SettingsGPS.model(GPS_VEHICLE); // Указываем модулю, что он используется в автомобиле.
```

Функция version();

- Назначение: Выбор версии протокола NMEA 0183 для формирования сообщений.
- Синтаксис: version(ВЕРСИЯ);
- Параметр:
  - float ВЕРСИЯ - может принимать одно из следующих значений: 2.2, 4.0, 4.1
- Возвращаемое значение: Нет.
- Примечание:
  - По умолчанию модуль формирует сообщения в соответствии с протоколом NMEA 0183 версии 4.1
- Пример:

```
SettingsGPS.version(4.1); // Указываем модулю формировать сообщения в соответствии с протоколом NMEA 0183 версии 4.1.
```

Функция reset();

- Назначение: Перезагрузка модуля.
- Синтаксис: reset(СТАРТ);
- Параметр:
  - uint8_t СТАРТ - определяет тип старта модуля после перезагрузки, может принимать одно из следующих значений:
    - GPS_HOT_START - «горячий старт» *стартовать с сохранением данных о спутниках*.
    - GPS_COLD_START - «холодный старт» *стартовать со сбросом данных о спутниках*.
    - GPS_FACTORY_SET - «холодный старт» *с восстановлением заводских настроек*.
- Возвращаемое значение: Нет.
- Примечание:
  - После перезагрузки модуль теряет все ранее рассчитанные данные, но сохранив данные о спутниках (при горячем старте) может значительно быстрее рассчитать данные заново.
- Пример:

```
SettingsGPS.reset(GPS_HOT_START); // Перезагрузить модуль с горячим стартом.
```

Функция save();

- Назначение: Сохранение текущих настроек модуля в его энергонезависимую память.
- Синтаксис: save();
- Параметры: Нет.
- Возвращаемое значение: Нет.
- Примечание:
  - В Flash память модуля сохраняются следующие настройки: скорость передачи данных, частота обновления выводимых данных, версия протокола NMEA и его состав, используемые спутниковые навигационные системы и динамическая модель.
  - Flash память модуля является энергонезависимой, а значит настройки сохранятся и после отключения питания, даже если у модуля не установлена батарейка.
  - Восстановить заводские настройки можно функцией `reset(GPS_FACTORY_SET)` .
- Пример:

```
SettingsGPS.save(); // Сохранить текущие настройки модуля в его энергонезависимую память.
```

Описание функций библиотеки iarduino_GPS_NMEA:

Библиотека iarduino_GPS_NMEA позволяет получать данные из текстовых сообщений NMEA 0183 отправляемых GPS-модулем по шине UART. Библиотека может использовать как аппаратную, так и программную реализацию шины UART для получения данных от GPS-модуля.

С подробным описанием функций библиотеки и примерами её работы можно ознакомиться на странице Wiki - Парсер протокола NMEA.

Список функций библиотеки:

Библиотека содержит 4 функции:

- **begin()** - *инициализация получения данных из строк NMEA.*
- **timeZone()** - *установка / получение часовой зоны.*
- **timeOut()** - *ограничение времени чтения данных.*
- **read()** - *чтение данных.*

Чтение данных:

Обращение к функции read() приводит к чтению данных из текстового сообщения NMEA 0183 отправляемого GPS-модулем по шине UART в следующие переменные:

- float **latitude** - Широта (±90.0°).
- float **longitude** - Долгота (±180.0°).
- uint16_t **altitude** - Высота над уровнем моря (±32767м).
- uint8_t **speed** - Скорость (0-255 км/ч).
- float **course** - Курс (±180.0°).
- uint8_t **satellites[**GPS_ACTIVE**]** - Количество активных спутников (0-12).
- uint8_t **satellites[**GPS_VISIBLE**]** - Количество наблюдаемых спутников.
- float **PDOP** - Пространственный геометрический фактор ухудшения точности (0-25.5).
- float **HDOP** - Горизонтальный геометрический фактор ухудшения точности (0-25.5).
- float **VDOP** - Вертикальный геометрический фактор ухудшения точности (0-25.5).
- uint8_t **seconds** - Секунды (0-59).
- uint8_t **minutes** - Минуты (0-59).
- uint8_t **hours** - Часы (1-12).
- uint8_t **Hours** - Часы (0-23).
- uint8_t **midday** - Полдень (0-am, 1-pm).
- uint8_t **day** - День месяца (1-31).
- uint8_t **weekday** - День недели (0-воскресенье, 1-понедельник, ... , 6-суббота).
- uint8_t **month** - Месяц (1-12).
- uint8_t **year** - Год (0-99).
- uint16_t **Year** - Год (0-65'535).
- uint32_t **Unix** - Unix время (0-4'294'967'296 сек).
- Функцию read() можно вызвать с указанием массива в качестве параметра, тогда этот массив будет заполнен данными о наблюдаемых спутниках.

Содержание данных в строках сообщения NMEA 0183:

Переменные:	**Идентификаторы строк сообщения NMEA 0183 отправляемого модулем:**								
latitude longitude	GGA	GLL	RMC	-	-	-	-	-	-
altitude	GGA	-	-	-	-	-	-	-	-
satellites	GGA	-	-	-	-	-	-	-	-
speed course	-	-	RMC	VTG	-	-	-	-	-
PDOP VDOP	-	-	-	-	-	GSA	-	-	-
HDOP	GGA	-	-	-	-	GSA	-	-	-
Hours hours minutes seconds midday	GGA	GLL	RMC	-	ZDA	-	-	DHV	GST
day weekday month year Year	-	-	RMC	-	ZDA	-	-	-	-
Unix	(GGA || GLL || RMC || ZDA || DHV || GST) && (RMC || ZDA)								
Массив	-	-	-	-	-	GSA	GSV	-	-

Наличие в составе сообщения NMEA 0183, отправленного модулем, строки с любым из указанных в таблице идентификатором, приводит к

заполнению данными соответствующей переменной.

Ссылки:

- Wiki - Парсер протокола NMEA.
- Wiki - Описание протокола NMEA 0183.
- Библиотека iarduino_GPS_ATGM336.
- Библиотека iatduino_GPS_NMEA.
- Wiki - Установка библиотек в Arduino IDE.