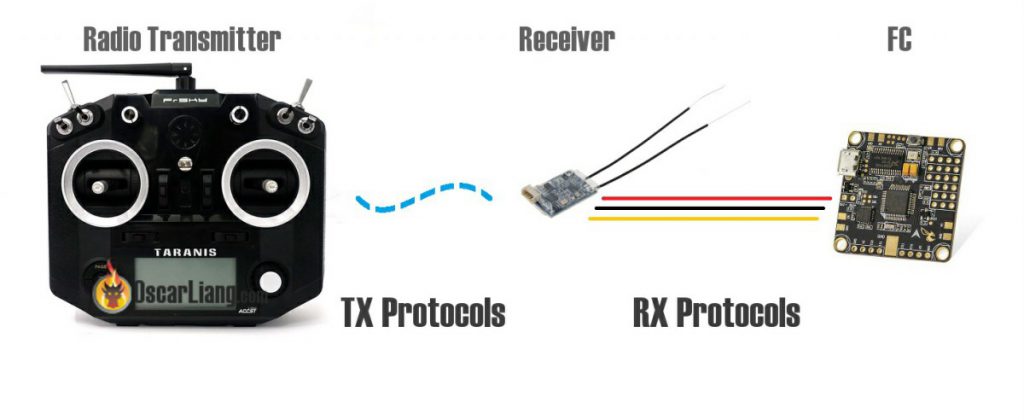
1. **Развитие прошивок для квадрокоптеров**
2. **Протоколы**



*Рисунок 1 – Мост передачи данных «Пульт – Полетный контроллер»*

В роли передатчика у нас выступает пульт, в роли приемника, черная коробочка, которая так и называется — приемник, он идет в комплекте с пультом и именно его вы устанавливаете в квадрокоптере. Приемник обрабатывает команды с пульта и отправляет их полетному контроллеру.

Протоколы можно разбить на 2 группы:

* Протоколы, применяемые в приемниках (связь между приемником сигнала управления и полетным контроллером) – **RX Protocols**
* Протоколы в передатчиках (обмен данными между передатчиком и приемником) - **TX Protocols**

Таблица 1 – Примеры протоколов

|  |  |
| --- | --- |
| Примеры TX протоколов | Примеры RX протоколов |
| D8 (Frsky)  D16 (Frsky)  LR12 (Frsky)  DSM (Spektrum)  DSM2 (Spektrum)  DSMX (Spektrum)  Flysky  A-FHSS (Hitec)  FASST (Futaba)  Hi-Sky (Deviation) | PWM (универсальный)  PPM (универсальный)  PCM (универсальный)  SBUS (Futaba, Frsky)  IBUS (Flysky)  XBUS (JR)  MSP (Multiwii)  SUMD (Graupner)  SUMH (Graupner)  CRSF – Crossfire (TBS)  FPort (Frsky) |

Таблица 2 – Сравнение протоколов «Receiver – FC»

|  |  |
| --- | --- |
| Протоколы | Описание |
| PWM (ШИМ) | Один канал/кабель для каждой сервомашинки. Мультикоптерам нужно как минимум 4 или 5 каналов (иногда даже больше), и именно такое же количество разъемов будет использоваться для подключения приемника к полетному контроллеру. |
| PPM – Pulse Position Modulation  (PPMSUM, CPPM) | Посылка ШИМ сигнала по одному проводу; свой сигнал на канал, идущие друг за другом (как правило до 8 каналов). Требует только 3 провода: питание, земля и сигнал. Передача данных немного не точная и есть джиттер (отклонение во времени) в отличие от цифрового последовательного канала. |
| PCM — Pulse Code Modulation | Импульсно кодовая последовательность, похожая на PPM, только это цифровой сигнал. PCM сигнал можно проверить на наличие ошибки, или даже использовать коды для исправления ошибок (но это зависит от устройств, в которых он применяется). |
| SBUS (S.BUS) | Протокол последовательной передачи данных, используемый Futaba и FrSky, который поддерживает передачу до 18 каналов по 1 проводу. SBUS — это инвертированный сигнал UART (COM порт). Поэтому для работы с ним требуется FC, имеющий встроенный инвертор сигнала. Используется Futaba и FrSky. |
| IBUS | Передача данных последовательная двунаправленная, т.е. можно как принимать, так и передавать данные — передавать данные к сервам и получать данные с датчиков. Используется FrSky. |
| XBUS | Последовательный XBUS используется фирмой JR, поддерживает до 14 каналов по 1 проводу. Одно из преимуществ — очень маленькая задержка сигнала между каналами. |
| MSP (multiwii serial protocol) | Последовательный протокол созданный как часть программного обеспечения multiwii. Позволяет передавать до 8 каналов по одному проводу. |
| Graupner Hott SUMD | Последовательный протокол типа SBUS. Каналы кодируются в один цифровой сигнал, и не имеет заметной задержки при передаче. Преимущества SUMD:  -по сравнению с SBUS — не требуется инвертор.  -по сравнению с PPM — выше разрешение сигнала, при этом PPM имеет всего 250 значений сигнала и дрожание (джиттер) сигнала = 4 мс. |
| Graupner SumH | Старый протокол от Graupner. Они выпустили обновление для своих приемников, чтобы можно было использовать SUMD вместо SumH. |
| CRSF (Crossfire) | Фирма TBS разработала этот протокол для своей аппаратуры Crossfire. Это цифровой протокол с принципом работы как SBUS, но у него более быстрая скорость работы, позволяющая использовать Телеметрию без использования доп.портов. По этому протоколу даже можно настраивать полетный контроллер. |
| FPort — марки Frsky и Betaflight | Новейший протокол связи от разработчиков Frsky и Betaflight, направленный на еще более быструю работу и максимальную компактность, для работы принятия и обработки всех сигналов телеметрии и управления, ему нужен только 1 провод. Протокол совместим с полетниками F4 без аппаратных инверторов. |

В аппаратуре Spektrum, протокол передачи данных от приемника к FC (копия SBUS) для Spektrum DSM2 называется SPEKTRUM1024, а для DSMX — SPEKTRUM2048.

Таблица 3 – Сравнение протоколов «Transmitter - Receiver»

|  |  |
| --- | --- |
| Spektrum DSM2 | Хорошо защищенный от помех, интерференции и от сигналов других передатчиков, работающих на той же частоте. Кроме того, при включении радиопередатчик ищет запасную частоту, на случай, если будут проблемы на основной. Это снижает вероятность потери сигнала, однако, если обе частоты будут заняты или станут слишком шумными, то сигнал будет потерян. |
| Spektrum DSMX | Улучшенная версия DSM2, использует ту же схему кодирования данных. Разница в том, что в случае DSMX, передатчик может за несколько миллисекунд перейти на новую частоту в случае проблем на основной, так что, теоретически, вы даже не заметите сбоя. |
| Spektrum Satellite | Это дополнительная антенна и приемник, который обычно подключается к основному приемнику и используется для увеличения надежности приема сигнала, позволяя выбрать антенну с наилучшим сигналом. |
| Протоколы Frsky | Типы радиомодуля:  -XJT (режимы D8, D16 и LR12, внутренний радиочастотный модуль для Taranis, внешний модуль типа JR для других радиостанций)  -XFT (внешний модуль XJT для радиостанций Futaba)  -DJT (только устаревшие протоколы D8, внешний модуль типа JR)  -DHT (D8 только устаревшие протоколы)  Приемники:  -D16: для приемников X-серии (X4R-SB, XSR, X8R)  -D8: для приемников D- и V-серий (D4R-II, D8R-II+, V8FR-II, VD5M и т.д.)  -LR12: для дальнобойной аппаратуры L9R |

**DShot** — это новый протокол обмена данными между полетным контроллером и регуляторами скорости (ESC), замена протоколам Oneshot и Multishot. DShot сокращение от Digital Shot.

Есть 3 варианта протокола, цифры определяют скорость интерфейса:

DShot600 – 600,000 бит/сек

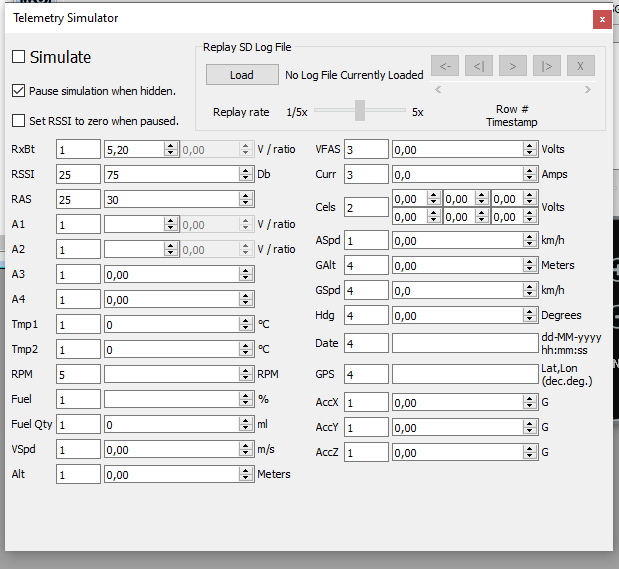
DShot300 – 300,000 бит/сек

DShot150 – 150,000 бит/сек

**Oneshot125** это более быстрый (чем обычный PWM сигнал) протокол передачи данных от полетного контроллера к регулятору. Он позволяет контроллеру чаще отсылать команды регулятору, что, теоретически, делает управление коптером более точным. Чтобы использовать Oneshot — нужно включить его как в настройках регуляторов, так и в настройках контроллера.

**Active Braking, Damped Light, DampingLight** - все это названия одной и той же функции регуляторов — активное торможение. Эта функция относится только к регуляторам, ее можно включить или выключить. Никакого отношения к настройкам полетных контроллеров она не имеет. Что это такое? Когда вы уменьшаете газ, регулятор будет активно замедлять двигатель. Если эта функция отключена — двигатель будет вращаться по инерции, и только воздух будет его притормаживать.

1. **Телеметрия**
   1. **Телеметрия в симуляторе OpenTX**



*Рисунок 2 – Симулятор телеметрии в программе Open TX Companion 2.3*

Таблица 4 – Расшифровка параметров в симуляторе телеметрии

|  |  |  |
| --- | --- | --- |
| Параметр | Расшифровка | Примечание |
| RxBt | напряжение аккумуляторной батареи приемника |  |
| RSSI | индикатор уровня принимаемого сигнала |  |
| A1 A2 | Аналоговое напряжение (приемник) |  |
| A3 A4 | Аналоговое напряжение (S.port в UART хост) |  |
| Tmp1 | барометрическая температура, если таковая имеется, в противном случае – гироскоп  Управление / режим полета | Не менять с цельсия на фаренгейт. Это штука используется для режимов полета и GPS |
| Tmp2 | Статус GPS и количество спутников (как num\_sats \* 10 + статус) |
| RPM | в режиме охраны: значение дроссельной заслонки, в противном случае - емкость аккумулятора. |  |
| Fuel | Оставшийся процент заряда батареи, если установлена емкость, в противном случае потребляется мАч |  |
| VSpd | вертикальная скорость |  |
| Alt | высота над уровнем моря по барометру, начальный уровень равен нулю |  |
| VFAS | Фактическое значение Vbat  Напряжение батареи |  |
| Curr | фактическое потребление тока, амперы  Потребление тока |  |
| Cels | среднее значение ячейки, деленное на количество ячеек. |  |
| ASpd | истинная скорость воздуха от датчика Пито. |  |
| GAlt | Высота по GPS, уровень моря нулевой.  Высота по GPS |  |
| GSpd | текущая горизонтальная путевая скорость, рассчитанная по GPS.  GPS скорость |  |
| Hdg | курс, север - 0 °, юг - 180 °  Угол рыскания |  |
| Date | время с момента включения |  |
| GPS | GPS координаты |  |
| AccX | Значения акселерометра | Акселерометр предоставляет данные:  Угол крена, угол наклона и G-Force X, Y, Z |
| AccY |
| AccZ |

В настоящее время поддерживаются несколько провайдеров телеметрии: FrSky, Graupner HoTT V4, SmartPort (S.Port) и LightTelemetry (LTM).

* 1. **Телеметрия SmartPort (S.Port)**

Используется передатчиками FrSky Taranis Q X7, X9D, X9D +, X9E или XJR в паре с приемниками серии X, такими как X4R (SB), X8R, XSR, R-XSR, XSR-M или XSR-E.

Устройства SmartPort используют инвертированный последовательный протокол и поэтому не могут быть напрямую подключены ко всем полетным контроллерам. Скорость передачи данных: 57600 бит/сек.

Могут передаваться следующие параметры с датчиков:

Таблица 5 – Доступные SmartPort датчики

|  |  |  |
| --- | --- | --- |
| Параметр | Значение | Примечание |
| GSpd | Текущая горизонтальная путевая скорость, рассчитанная с помощью GPS |  |
| VFAS | Фактическое значение напряжения vbat |  |
| Curr | Фактический потребляемый ток в амперах |  |
| Alt | Барометрическая высота относительно домашнего местоположения |  |
| Fuel | Если smartport\_fuel\_unit = PERCENT шлется оставшийся процент заряда батареи, иначе мА/ч |  |
| GPS | GPS координаты |  |
| VSpd | Вертикальная скорость, см/с |  |
| Hdg | Курс, север - 0°, юг - 180° |  |
| AccX,Y,Z | Данные акселерометра (не отправляются, если frsky\_pitch\_roll = ON) |  |
| Tmp1 | Полетный режим, отправляется 5 цифр. Номер отправляется как ABCDE. Цифры суммируются, например, если цифра С = 6 – это состояния двух режимов: удержания высоты (2) и удержания положения (4) | A:  1 – режим флаперона  (flaperon mode)  2 – режим автонастройки  (auto tune mode)  4 - отказоустойчивый режим (failsafe mode)  B:  1 - возврат домой (return to home)  2 - режим путевой точки (waypoint mode)  4 - режим без головы (headfree mode)  C:  1 - удержание курса (heading hold)  2 - удержание высоты (altitude hold)  4 - удержание положения (position hold)  D:  1 - режим угла (angle mode)  2 - режим горизонта (horizon mode)  4 - режим прохода (passthru mode)  E:  1 - постановка на охрану разрешена (ok to arm)  2 - постановка на охрану запрещена (arming is prevented)  4 - постановка на охрану (armed) |
| Tmp2 | Состояние блокировки GPS, точность, триггер возврата в исходное состояние и количество спутников. Номер отправляется в формате ABCD. Обычный минимальный GPS блок составляет 3906: GPS заблокирован и дом зафиксирован, HDOP (Horizontal Dilution of Precision - снижение точности в горизонтальной плоскости) высочайшая точность, 6 спутников). | A:  1 - определение местоположения по GPS  2 - исходное положение по GPS  4 - сброс исходного состояния (числа складываются)  B:  1 - точность GPS на основе HDOP (от 0 = наименьшая до 9 = наибольшая точность)  C:  количество заблокированных спутников (цифры C и D - количество заблокированных спутников)  D:  количество заблокированных спутников (если заблокировано 14 спутников, C = 1 и D = 4) |
| GAlt | Высота по GPS над уровнем моря |  |
| ASpd | Истинная скорость воздуха от датчика Пито |  |
| A4 | Среднее значение ячейки.  Предупреждение: в отличие от датчиков FLVSS (Датчик напряжения FrSky FLVSS (smart порт) для LiPo батарей) и MLVSS (FrSky Mini Lipo Voltage Sensor MLVSS), вы не получаете фактическое наименьшее значение ячейки, а получаете среднее значение: (общее напряжение литий-полимерного аккамулятора) / (количество ячеек) |  |
| 0420 | Дистанция по GPS до зафиксированного дома |  |
| 0430 | если frsky\_pitch\_roll = ONустановлено, это будет шаг градусов \* 10 |  |
| 0440 | если frsky\_pitch\_roll = ONустановлено, это будет градус крена \* 10 |  |
| 0450 | «Вектор траектории полета» или «Курс относительно земли» в градусах \* 10 |  |

Настройка UART:

* 57600
* 8 бит
* 1 стоп бит
* нет бита четности
* полудуплекс
* активный уровень инвертирован

Протокол FPort – это телеметрия Smart Port (S.Port) + SBUS.

Здесь процесс получения телеметрии состоит в следующем:

Приемник (в данном случае, FrSky R-XSR) отправляет запрос по UART вида:

- заголовок 0x7E

- 8 битный id датчика

Если датчик присутствует – он отвечает отправкой сообщения вида:

- заголовок кадра данных 0x10

- 16 битный тип параметра (например, напряжение / скорость) value\_type

- 32 битное значение параметра (со знаком или без в зависимости от типа самого значения)

- 8 битная контрольная сумма

Если датчик отсутствует и ответ не отправляется – происходит запрос датчика со следующим id.

Если два датчика используют один и тот же идентификатор, шина не будет работать, так как оба датчика будут пытаться отправить данные одновременно. Это приведет к разрыву телеметрического соединения.

Поток данных запроса телеметрии по S.Port от приемника R-XSR – это 28 байт с id датчиков, перед каждым id идет заголовок 0x7E (всего 56 байт).

Таблица 6 – ID датчиков S.Port

|  |  |  |  |
| --- | --- | --- | --- |
| Физический ID | ID | Датчик | Описание |
| 0 | 0x00 | Vario2 | Высотометр высокой точности |
| 1 | 0xA1 | FLVSS | Датчик батареи (может быть отправлен с одним или двумя напряжениями ячеек) |
| 2 | 0x22 | FAS-40S | Датчик тока |
| 3 | 0x83 | GPS / высотомер | Нормальная точность |
| 4 | 0xE4 | RPM | Оборот / мин |
| 5 | 0x45 | SP2UART (Host) | Преобразователь Smart Port в UART, хост |
| 6 | 0xC6 | SP2ART (Remote) | Преобразователь Smart Port в UART, удаленный |
| 7 | 0x67 |  |  |
| 8 | 0x48 |  |  |
| 9 | 0xE9 | ASS | Воздушная скорость |
| 10 | 0x6A |  |  |
| 11 | 0xCB |  |  |
| 12 | 0xAC |  |  |
| 13 | 0x0D |  |  |
| 14 | 0x8E |  |  |
| 15 | 0x2F |  |  |
| 16 | 0xD0 |  |  |
| 17 | 0x71 |  |  |
| 18 | 0xF2 |  |  |
| 19 | 0x53 |  |  |
| 20 | 0x34 |  |  |
| 21 | 0x95 |  |  |
| 22 | 0x16 |  |  |
| 23 | 0xB7 | IMU ACC (x, y, z) | Инерциальный измерительный модуль Акселерометр (3 оси) |
| 24 | 0x98 | RX / TX telemetry | Прием / передача телеметрии |
| 25 | 0x39 | Power Box | Блок питания |
| 26 | 0xBA | Temp |  |
| 27 | 0x1B | Fuel | Топливо (ArduPilot / Betaflight) |

Получатель, похоже, не смотрит на ответ, чтобы убедиться, что он относится к тому же типу запрошенных данных, поэтому оказывается, что датчик может реагировать на любой запрос от получателя с любым типом данных.

* 1. **FrSky телеметрия**

Телеметрия FrSky предназначена для старых передатчиков FrSky и приемников серии D. Для более новых передатчиков, сопряженных с приемниками серии X, см. Телеметрию SmartPort (S.Port) выше. Телеметрия FrSky предназначена только для передачи и требует всего лишь подключения одного контакта TX последовательного порта к контакту RX на приемнике телеметрии FrSky. Сигналы телеметрии FrSky инвертированы. Чтобы подключить плату с поддержкой INAV к приемнику FrSKy, у вас есть несколько вариантов:

-Аппаратный инвертор - встроен в некоторые полетные контроллеры.

-Используйте программный последовательный порт.

-Используйте полетный контроллер с программно конфигурируемой аппаратной инверсией (например, F3 или F7).

Таблица 6 – Доступные параметры FrSky S-Port

|  |  |  |
| --- | --- | --- |
| Параметр | Значение | Примечание |
| AccX, AccY, AccZ | Акселерометр |  |
| Alt | Высота по барометру |  |
| Curr | Фактическое потребление тока (Амперы) |  |
| Fuel | оставшийся процент заряда батареи, если battery\_capacity переменная установлена ​​и переменная smartport\_fuel\_percent = ON, в противном случае - мАч. |  |
| GAlt | высота по GPS, нулевой уровень моря. |  |
| GPS | координаты GPS |  |
| GSpd | Текущая горизонтальная путевая скорость, рассчитанная с помощью GPS |  |
| Hdg | направление (градусы - север 0 °) |  |
| VFAS | Фактическое значение напряжения батареи (Voltage FrSky Ampere Sensor) |  |
| VSpd | Вертикальная скорость (см / с) |  |
| Tmp1 | Режим полета | передается как целое число:  18 - Ручной,  23 - Высота,  22 - Положение,  27 - Миссия,  26 - Удержание,  28 - Возвращение,  19 - Акро,  24 0 За бортом,  20 - Стабилизировано, 25 - Взлет,  29 - Земля,  30 - Следуй за мной. |
| Tmp2 | информация GPS | Крайняя правая цифра обозначает тип местоположения GPS (0 = нет, 2 = 2D, 3 = 3D). Остальные цифры - это количество спутников |

Таблица 7 – Доступные параметры FrSky D-Port

|  |  |  |
| --- | --- | --- |
| Параметр | Значение | Примечание |
| AccX, AccY, AccZ | Акселерометр |  |
| Alt | Высота по барометру |  |
| Cels | Среднее значение напряжения ячейки (напряжение батареи, деленное на количество ячеек). |  |
| Curr | Фактическое потребление тока (Амперы) |  |
| Fuel | оставшийся процент заряда батареи, если battery\_capacity переменная установлена ​​и переменная smartport\_fuel\_percent = ON, в противном случае - мАч. |  |
| Date | Время с момента включения |  |
| GAlt | высота по GPS, нулевой уровень моря. |  |
| GPS | координаты GPS |  |
| GSpd | Текущая горизонтальная путевая скорость, рассчитанная с помощью GPS |  |
| Hdg | направление (градусы - север 0 °) |  |
| RPM | значение дроссельной заслонки в режиме охраны, в противном случае емкость аккумулятора. Обратите внимание, что номер лезвия в Таранисе должен быть установлен на 12. |  |
| Tmp1 | Режим полета | передается как целое число:  18 - Ручной,  23 - Высота,  22 - Положение,  27 - Миссия,  26 - Удержание,  28 - Возвращение,  19 - Акро,  24 0 За бортом,  20 - Стабилизировано, 25 - Взлет,  29 - Земля,  30 - Следуй за мной. |
| Tmp2 | информация GPS | Крайняя правая цифра обозначает тип местоположения GPS (0 = нет, 2 = 2D, 3 = 3D). Остальные цифры - это количество спутников |
| VFAS | Фактическое значение напряжения батареи (Voltage FrSky Ampere Sensor) |  |
| VSpd | Вертикальная скорость (см / с) |  |

* 1. **HoTT телеметрия**

Эмулируются только электрические воздушные модули и модули GPS.

Используйте последнюю версию прошивки Graupner для вашего передатчика и приемника.

Более старые передатчики HoTT требовали, чтобы модули EAM и GPS были включены в меню телеметрии передатчика. (например, на MX-20)

* 1. **LightTelemetry (LTM)**

LTM - это облегченный протокол потоковой телеметрии, поддерживаемый рядом экранных меню, наземных станций и антенных трекеров.

Реализация LTM в INAV реализует следующие фреймы:

G-FRAME: информация GPS (широта, долгота, путевая скорость, высота, информация о спутнике)

A-FRAME: Отношение (тангаж, крен, курс)

S-FRAME: Статус (напряжение, ток +, RSSI, воздушная скорость +, статус). Элемент с суффиксом «+» не реализован в INAV.

O-КАДР: исходная точка (исходное положение, широта, долгота, высота, привязка)

Кроме того, в iNav:

N-FRAME: навигационная информация (режим GPS, режим навигации, действие навигации, номер маршрутной точки, ошибка навигации, флаги навигации).

X-FRAME: Дополнительная информация. В настоящее время сообщается о HDOP.

LTM работает только на передачу и может работать с любой поддерживаемой скоростью передачи. Он разработан для работы со скоростью более 2400 бод (9600 в INAV) и не имеет преимуществ от более высоких скоростей. Таким образом, его можно использовать в мягких сериалах.

* 1. **Телеметрия MAVLink**

MAVLink - это очень легкая библиотека для сортировки сообщений только для заголовков для микровоздушных транспортных средств. INAV поддерживает MAVLink для совместимости с наземными станциями, экранными меню и антенными трекерами, созданными для платформ PX4, PIXHAWK, APM и Parrot AR.Drone.

Реализация MAVLink в INAV предназначена только для передачи и может использоваться на низких скоростях передачи и может использоваться через программный последовательный порт.

* 1. **Сотовая телеметрия с помощью текстовых сообщений**

INAV может использовать сотовый модуль SimCom SIM800 для передачи телеметрии с помощью текстовых сообщений. Сообщения телеметрии можно запросить, позвонив на номер модуля или отправив ему текстовое сообщение. Модуль может быть настроен на передачу сообщений через равные промежутки времени или при обнаружении события ускорения. Команду текстового сообщения можно использовать для перевода полетного контроллера в режим возврата домой.

* 1. **Телеметрия IBUS**

Телеметрия IBUS требует одного соединения сигнала TX двунаправленного последовательного порта с ножкой датчика IBUS sens приемника телеметрии FlySky (проверено на приёмниках fs-iA6B и iA10).

Телеметрия Ibus включена по умолчанию во всех прошивках.

Доступные датчики:

1. Внутреннее напряжение в вольтах (не используется).

2. Датчик напряжения в вольтах (тип напряжения).

3. Если имеется барометрический датчик, то температура обратного потока от барометрического датчика в °C, иначе температура обратного потока от гироскопического датчика в °C (тип Temperatyre).

4. Статус (тип об/мин).

5. Курс в градусах (тип об/мин).

6. Ток в амперах (тип напряжения).

7. Высота в метрах (тип напряжения).

8. Направление к дому в градусах (тип об/мин).

9. Расстояние до дома в метрах (тип об/мин).

10. Курс GPS в степени (тип об/мин).

11. Высота GPS в метрах (тип об/мин).

12. Вторая часть широты (тип об/мин), например 5678 (-12,3456789 Н).

13. Вторая часть долготы (тип об/мин), например 6789 (-123,4567891 E).

14. Первая часть широты (тип напряжения), например -12,45 (-12,3456789 Н).

15. Первая часть долготы (тип напряжения), например -123,45 (-123,4567890 E).

16. Скорость GPS в км/ч (тип об/мин).

1. Напряжение передатчика в вольтах (не используется).

1. Процент ошибок в% (не используется).

Датчики от 8 до 16 доступны только при наличии GPS на момент сборки.

СОСТОЯНИЕ (количество спутников AS # 0, FIX AS 0, HDOP AS 0, Mode AS 0)

ИСПРАВЛЕНИЕ: 1 - нет, 2 - 2D, 3 - 3D, 6 - нет + FixHome, 7 - это 2D + FixHome, 8 - это 3D + FixHome

HDOP: 0 - 0-9 м, 8 - 80-90 м, 9 -> 90 м.

Режим: 0 - Passthrough, 1-Armed (скорость), 2-Horizon, 3-Angle, 4-Waypoint, 5-AltHold, 6-PosHold, 7-Rth, 8-Launch, 9-Failsafe

Пример: 12803 - это 12 спутников, Fix3D, FixHome, 0-9 м HDOP, угловой режим.

1. **Пульт. Модули расширения**

В пульте используется прошивка OpenTX.

Для большего охвата по каналам, в пульт можно добавить модуль передатчика типа JR (внешний, вставляющийся снизу в специальный отсек). Выглядит он как-то так:



*Рисунок 3 – Вид с лицевой стороны внешнего модуля типа JR для пульта*



*Рисунок 4 – Вид с задней стороны внешнего модуля типа JR для пульта*

Как видно, на задней стороне размещена 5 гнездовая розетка, в пульте в отсеке для этого модуля находится 5 контактная вилка.

В самом пульте стоит модуль (внутренний) XJT, который совместим с приемниками серий X, D, V8-II.

Данный пуль поддерживает протокол ACCESS (Advanced Communication Control Elevated Spread Spectrum - усовершенствованный контроль связи с расширенным спектром распространения).

ACCESS - самый продвинутый протокол FrSky на сегодняшний день. Основываясь на известном протоколе ACCST, ACCESS предлагает новые функции, такие как Smart Share, Smart Match и Trio Control. Как только приемник привязан к радио аппаратуре, его можно в любое время повторно привязать, не нажимая кнопку F/S на приемнике. Совместное использование моделей никогда не было проще. Обновление также было упрощено, вам больше не нужно физически подключать свои приемники для обновления. Кроме того, появились новые инструменты, такие как анализатор спектра и анализатор мощности. И самое главное, теперь протокол поддерживает до 24 каналов с повышенной скоростью передачи, меньшей задержкой и повышенной производительностью.

ОСОБЕННОСТИ ПРОТОКОЛА ACCESS:

-Поддерживает 24 широкополосных канала

-Увеличенная скорость передачи данных и меньшая задержка

-Беспроводная привязка, обновления и конфигурация

-Автоматический бинд (Smart Match)

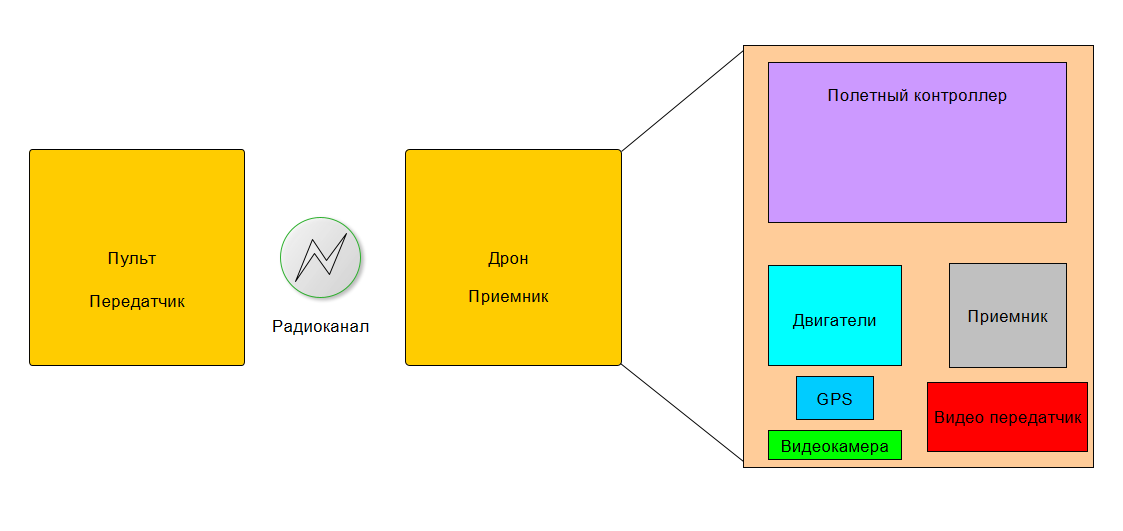
-Переназначение канала

-Управляемая телеметрия (Trio ControlTM)

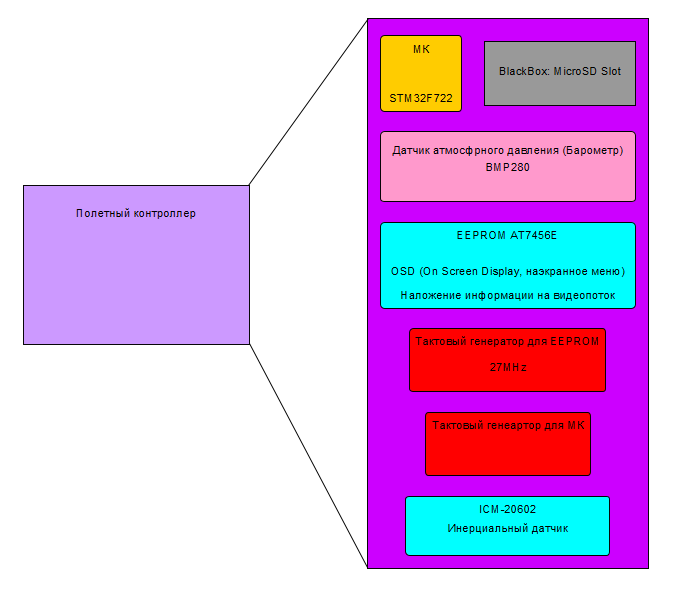
-Позволяет радио аппаратуре действовать как анализатор спектра

-Позволяет радио аппаратуре выступать в качестве измерителя мощности

-Расширенный алгоритм шифрования данных



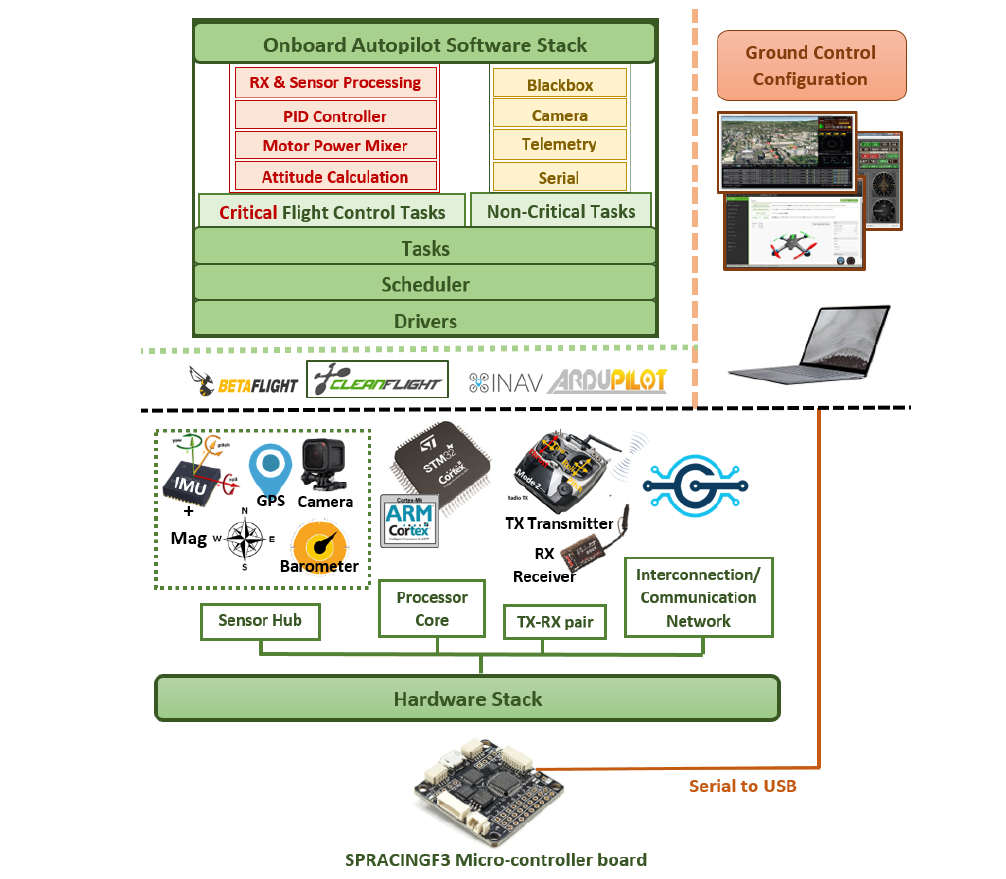
*Рисунок 5 – Система «Пульт – Дрон»*



*Рисунок 6 – Структура платы Полетного контроллера*

1. **Адаптивная система управления полетом в реальном времени**
   1. **Общие сведения**

В основе всех многороторных летательных аппаратов лежит автопилот. Это микропрограммное обеспечение или программное обеспечение полетного контроллера, которое объединяет обработку данных датчиков с оценкой ориентации 1 и регулировкой скорости ротора для поддержания целевой траектории. Управление полетом на типичных многороторных беспилотных летательных аппаратах использует классические линейные механизмы, которые неэффективны по энергии, недетерминированы во времени и подвержены нестабильности. Современные автопилоты не обладают способностью динамически адаптировать поведение в полете в ответ на внешние помехи. Следовательно, меняющиеся условия окружающей среды, такие как ветер, значительно влияют на способность дрона поддерживать полет по заданной траектории. Часто требуется ручное вмешательство для исправления ситуаций, которые в противном случае привели бы к авариям или невозможности достижения целей полета.



*Рисунок 7 - Аппаратные и программные компоненты «прошивки» автопилота. Критические и некритические задачи отправляются на выполнение базовым планировщиком. Приложения конфигурации обычно запускаются на наземных компьютерах для начальной настройки различных параметров, таких как параметры задания, время контура, усиление ПИД-регулятора и частоты среза фильтра.*

Ключевые проблемы, непосредственно влияющие на характеристики дрона, включают:

1) отсутствие адаптируемого и предсказуемого по времени управления полетом,

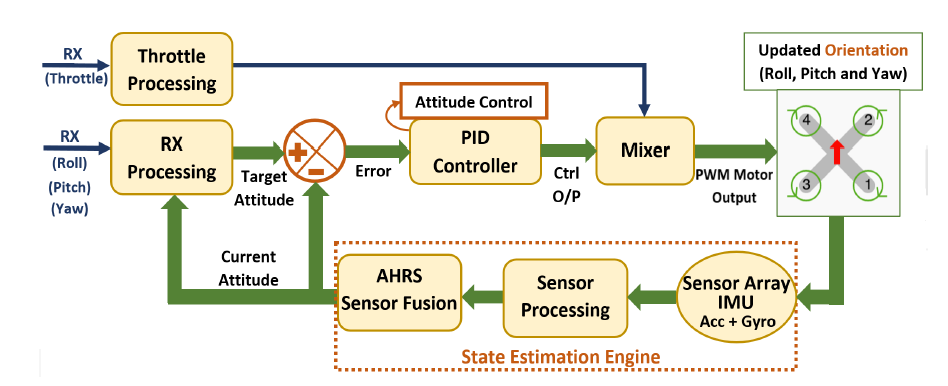
2) невозможность точного реактивного восстановления стабильного полета,

3) отсутствие восстановления ориентации с малой задержкой и быстрым временем отклика,

4) неэффективность использования ограниченного заряда батареи.

Прошивка полетного контроллера использует обработку данных датчиков и существующие функции управления полетом автопилота для автономного противодействия неблагоприятным воздействиям окружающей среды на положение дрона. Это достигается за счет разумной адаптации скорости выполнения критических задач полетного контроллера, гарантируя при этом их критические для безопасности ограничения планирования в реальном времени.

Современные системы автопилота сочетают в себе функции управления полетом, критичные для безопасности, с задачами, не критичными для безопасности, такими как обработка данных камеры для экранных дисплеев (OSD), передача данных телеметрии и запись данных черного ящика на накопитель. На рисунке 7 показан пример стека программного обеспечения, управляемого планировщиком, вместе с основными аппаратными модулями, общими для реальных архитектур коптеров с автопилотом.



*Рисунок 8 - Компоненты контура управления полетом автопилота*

Критические задачи управления полетом, отвечающие за стабилизацию ориентации на малых высотах, часто тесно связаны в замкнутом контуре. Эти задачи являются основными составляющими широко используемого метода линейного управления с отрицательной обратной связью, изображенного на рисунке 8. Цикл включает высокочастотную выборку для сбора и обработки данных от нескольких датчиков. Затем используется серия фильтров и сложных алгоритмов объединения датчиков для оценки текущей ориентации или положения дрона как части механизма оценки состояния. Эти данные сравниваются с целевым положением, полученным через модуль rx, и ошибка подается на ПИД-регулятор (пропорциональный, интегральный и дифференцирующий). Три регулятора преобразуют ошибку между фактическим и желаемым угловыми состояниями в управляющие сигналы для электронных регуляторов скорости (electronic speed controllers, ESC). Выходные команды контроллера, смешанные с входными данными дроссельной заслонки, затем используются для настройки индивидуальных скоростей двигателя и оборотов ротора в минуту (RPM). Это компенсирует фактические маневры коптера по сравнению с желаемыми. Комбинация дифференциальных угловых скоростей приводит к приложенному чистому крутящему моменту вокруг центра тяжести коптера, заставляя систему подвергаться поворотам на угол Эйлера, включая крен, тангаж и рыскание, по отношению к точке отсчета Земли.

Для облегчения управления полетом система оснащена:

1) сетью датчиков (например, акселерометрами и гироскопами, вместе известными как блок инерциальных измерений (IMU), магнитометрами, барометрами, сонарами, камерами, GPS и т. д.)

2) полным силовым агрегатом с двигателями, роторными винтами и ESC

3) подсистемой распределения и регулирования мощности.

В совокупности вся система обеспечивает стабильное и точное управление полетом в установившихся условиях. Однако, как упоминалось ранее, современные конструкции систем автопилота очень чувствительны к внешним изменениям в динамике полета и, таким образом, не работают, когда подвергаются переходным нарушениям ориентации. Стратегия планирования в реальном времени с адаптацией к скорости и критичностью решает проблему стабильности, дополняя существующие полетные контроллеры повышенным вниманием к окружающей среде. Он улучшает стабильность и характеристики полета за счет применения модульного и структурированного подхода для достижения детерминированного, предсказуемого по времени и адаптируемого управления полетом.

Политика адаптации скорости основана на ключевом понимании того, что производительность полетного контроллера напрямую связана со скоростью выполнения его критического контура управления полетом. Когда дрон часто меняет свою ориентацию, датчики с высокой частотой дискретизации собирают самые свежие данные, отображающие состояние дрона. Это позволяет нижележащему ПИД-регулятору последовательно корректировать текущую ошибку ориентации во всех трех навигационных измерениях, а именно по крену, тангажу и рысканью.

Таким образом, контроллер может внимательно отслеживать мгновенные силы, действующие на коптер в любой момент времени. Если внешние силы на коптер значительно различаются, то соответственно более высокая частота обновлений скоростей двигателя приводит к гораздо большей детализации и точности корректировок ориентации и управления. В качестве альтернативы, нечастая обработка данных датчиков и оценка состояния приводят к неточным прогнозам ориентации, что приводит к потенциально неправильным настройкам скорости двигателя для поддержания целевой траектории полета.

Напротив, стабильные погодные условия практически не вызывают аномальных изменений положения, что дает возможность снизить скорость выполнения контура управления. Это позволяет эффективно управлять вычислительными ресурсами, делая их доступными для других некритических функций миссии, таких как обнаружение и отслеживание объектов, обработка данных камеры для предотвращения препятствий и возможная навигация по путевым точкам. Бесщеточные двигатели постоянного тока, отвечающие за создание необходимой подъемной силы для поддержания полета, подключены непосредственно к источнику питания от основной батареи и являются основными потребителями энергии на дроне. Таким образом, уменьшение количества требуемых обновлений скорости двигателя обеспечивает энергоэффективное управление полетом.

Наша структура управления полетом:

1) определяет задачи как критические для безопасности и некритичные для безопасности в рамках хорошо известной системы автопилота,

2) определяет рабочие частоты для отдельных задач в соответствии с текущим режимом критичности системы,

3) самостоятельно рассматривает изменяющиеся внешние условия для динамической адаптации скорости выполнения задач принципиальным образом,

4) обеспечивает управление в реальном времени и учет циклов процессора,

5) гарантирует жестко установленные границы времени выполнения для всех задач полетного контроллера, чтобы гарантировать успешный полет при наличии временных неопределенностей.

ПО полетного контроллера компенсирует мгновенные изменения в окружающей среде в пределах предсказуемого времени, обеспечивая малую задержку реакции на критические внешние события, предотвращая сбои и ускоряя восстановление после аномальных изменений отношения. По сравнению с автопилотами со статической скоростью, тот, который используется в ПО полетного контроллера, позволяет избежать ненужных циклов управления в относительно стабильных погодных условиях, чтобы сэкономить заряд батареи и освободить вычислительные ресурсы для дополнительных задач.

Несмотря на различия, автопилоты (прошивки полетного контроллера) имеют более или менее аналогичную логику управления полетом, что и стандартный контур обратной связи, изображенный на рисунке 8.

smartflight работает по принципу общей применимости для всех прошивок автопилота, расширяя существующую логику управления с возможностью адаптации скорости выполнения задач во время выполнения. В качестве примера реализации мы модернизируем один из самых популярных настраиваемых автопилотов, Cleanflight (CF), с архитектурой планирования smartflight. Разработанный специально для гоночных квадрокоптеров, CF сохраняет конкурентное преимущество перед другими популярными полетными контроллерами с открытым исходным кодом с точки зрения эффективности полета, функциональной надежности и производительности контроллера. В сочетании с функционально надежным и минималистичным стеком управления полетом он оказывается идеальной платформой для автопилота, демонстрирующей преимущества в производительности smartflight на экономичном встроенном оборудовании с ограниченными ресурсами.

Несмотря на свои преимущества, Cleanflight не отличается гибкостью в работе. Задачи управления полетом определяются со статическими периодами времени, которые действуют только как мягкие временные границы. Базовый планировщик также основан на политике планирования с максимальным приоритетом, не требующей вытеснения. Отсутствие предсказуемости по времени является одним из основных факторов, влияющих на переменное время диспетчеризации задач, которое часто проявляется в том, что задание начинается раньше, чем ожидалось, или завершается позже, чем требуется. Различия в задержках между временем выпуска заданий для одной и той же задачи становятся серьезным препятствием для предсказуемой адаптации в системе. Таким образом, отсутствие строгих временных ограничений делает выполнение задачи недетерминированным, что отрицательно сказывается на безопасности и надежности управления полетом. Поэтому, чтобы избежать катастрофического отказа из-за внешних возмущений, время выполнения задач должно точно контролироваться. Мы проверяем эту неотъемлемую неопределенность во времени выполнения задач с помощью реальных экспериментов с Vanilla CF в качестве фазы I наших оценок. В следующем разделе дается описание реализации Vanilla CF.

* 1. **Задачи Cleanflight**

Таблица 8 - Список основных задач Cleanflight: (жирным шрифтом выделены критические задачи, связанные с управлением полетом на малых высотах).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя задачи | Временной период | Частота исполнения (Гц) | Статический приоритет | Критичность | Описание |
| task\_system | 100 000 | 10 | Med-High | Low | Сбор системной статистики |
| task\_bat\_volt | 20 000 | 50 | Medium | Low | Замер напряжения батареи |
| **task\_gyropid (время цикла)** | **4000 / 2000 / 1000** | **250 / 500 / 1000** | **Real-Time**  **(наивысший)** | **High** | **Замер гироскопа + ПИД-регулирование двигателями** |
| **task\_accel** | **1000** | **1000** | **Medium** | **High** | **Замер данных гироскопа** |
| **task\_attitude** | **10 000** | **100** | **Medium** | **High** | **Расчет текущей позиции** |
| task\_rx | 20 000 | 50 | High | Low | Прием команд |
| task\_serial | 10 000 | 100 | Low | Low | Последовательный обмен с ПК на земле |

В Cleanflight (v.2.3.1) всего 31 задача, более половины из которых являются опциональными дополнительными функциями. Важнейшие функции управления полетом распределены по набору из 3 задач: task\_gyropid, task\_accel и task\_attitude. Основной контур управления (рисунок 8) состоит из:

1) задачи приемника (task\_rx), которая обрабатывает эталонные входные данные для целевого положения крена, тангажа и рыскания,

2) отдельных ПИД-регуляторов для каждой оси вращения, которые регулируют выходную реакцию на основе на текущую ошибку ориентации (усиление P), накопление прошлых ошибок (усиление I) и скорость изменения ошибки (усиление D),

3) компонент смесителя, который определяет величину силы тяги, приложенной к каждому двигателю,

4) нескольких задач обработки датчиков (task\_accel и task\_gyropid),

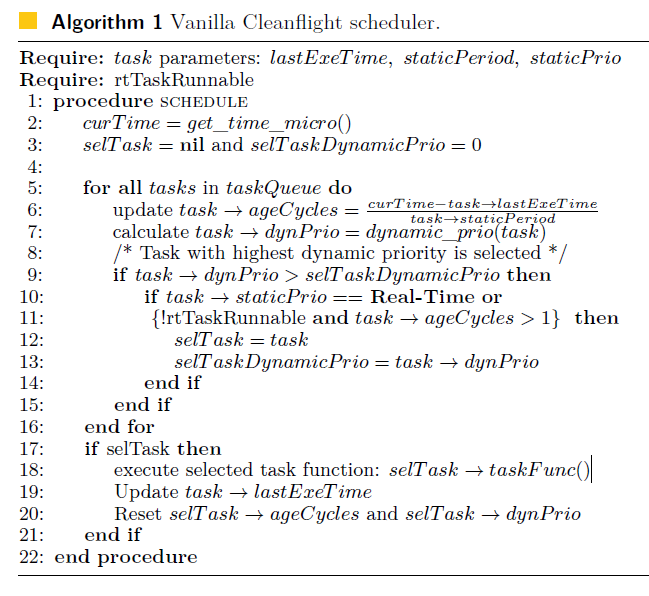
5) основанная на кватернионах подсистема привязки и направления (AHRS: task\_attitude), которая объединяет данные датчиков с использованием дополнительного алгоритма фильтрации Madgwick & Mahony для вычисления текущее положения коптера.

Обработка дроссельной заслонки включена в task\_rx. Радио или наземная станция управления передает команды по беспроводной сети, которые затем принимаются аппаратным модулем RX на дроне. task\_rx различается по своей важности для общего механизма управления полетом, в зависимости от частоты обновления целевых позиций, часто диктуемой приложением дрона.

Отметим, что task\_gyropid включает в себя цепочку подзадач, перечисленных ранее как пункты 2), 3) и 4) основного контура управления. Эта задача последовательно производит выборку данных гироскопа, выполняет алгоритм ПИД-регулятора и обновляет выходные команды двигателя. Таким образом, task\_gyropid образует быстрый цикл, который обеспечивает настраиваемую пользователем скорость выполнения, известную как базовое время цикла в терминологии CF. Таким образом, время цикла представляет собой период времени для одной итерации быстрого цикла управления. Другие критические задачи в основном цикле управления, а именно (task\_accel и task\_attitude), работают с неконфигурируемыми частотами, которые являются фиксированными целыми кратными базовой длительности цикла.

* 1. **Не модифицированный планировщик**

Алгоритм 1 детализирует политику не вытесняющего планирования Vanilla CF. Планировщик поддерживает фиксированную очередь готовности задач в порядке убывания статических приоритетов из таблицы 8. Задачи планируются от самого высокого до самого низкого динамического приоритета, который рассчитывается во время выполнения для каждой задачи как произведение ее статического приоритета и прошедшего времени, с момента последнего выполнения (возрастные циклы задачи, task age-cycles) (строки 6–7). Очередь просматривается при каждом вызове планировщика, и задача с наивысшим динамическим приоритетом отправляется на выполнение в соответствии со строками 9–11. Выбранная задача либо имеет статический приоритет в реальном времени, либо имеет более низкий статический приоритет и устарела как минимум в течение двух последовательных периодов времени.



Отметим, что в контексте Vanilla CF «в реальном времени» не накладывает никаких временных ограничений на задачу, а вместо этого используется для представления наивысшего статического приоритета. Отправленная задача выполняется до завершения и только совместно передает управление планировщику в конце своего выполнения. В зависимости от времени выполнения задачи время между последовательными вызовами планировщика может значительно различаться.

* 1. **Модель задачи**

Система моделируется из набора задача в реальном времени {t1, t2, …, tn}. В примере каждая задача ti имеет 5 параметров: {Ci, [Ti(LO), Ti(HI)], [Di(LO), Di(HI)], Li, [pi(LO), pi(HI)]}, где

Ci – время вычисления или бюджет. Вычислительная логика и структура задачи остаются неизменными. Это означает, что время выполнения также остается более или менее одинаковым для нескольких экземпляров одной и той же задачи. Мы определяем бюджет времени выполнения путем профилирования системы в режиме онлайн при каждом запуске системы. Это значение бюджета затем используется для расчета использования задачи (Ui = Ci/Ti), который затем используется для вычисления использования процессора (Usys).

Ci вычисляется пессимистично как верхняя граница фактического времени вычислений задачи путем интеграции всех возможных накладных расходов на прерывания, которые могут быть включены в бюджет времени выполнения задачи из-за запросов ввода-вывода и памяти, в дополнение к накладным расходам планировщика в пределах базового времени.

Ti = [Ti(LO), Ti(HI)] - вектор периодов времени. Каждая задача i имеет соответствующий период Ti для каждого уровня критичности в системе, где Lsys = {LO, HI} - набор уровней критичности системы. Задачи явно изменяют свои периоды времени при изменении режима системы. Каждый период времени является мультипликативной обратной величиной соответствующей скорости выполнения задач (Ri): Ti (Lsys) = 1 / (Ri (Lsys)).

1. **Управление двигателями**

Управление двигателями дрона происходит с помощью ESC (*Electronic Speed Controller*) – электронного регулятора хода. Протоколы управления (часто обмен симплексный «полетный контроллер -> esc») можно условно разделить на две категории: управление путем подачи ШИМ сигнала и управление с использованием цифрового протокола. К первой категории относятся следующие протоколы:

-STANDARD,

-ONESHOT125,

-ONESHOT42,

-MULTISHOT,

-BRUSHED.

Ко второй:

-DSHOT150,

-DSHOT300,

-DSHOT600,

-DSHOT1200

Инициализация ESC - это определение ESC через какой протокол с ним будут общаться. Для этого нужно некоторое время (порядка 2-5 секунд) подавать на ESC нулевое значение выбранного протокола. После определения протокола ESC должен издать мелодичный звук готовности работать.

Самый простой протокол управления – это устаревший «аналоговый» сигнал ШИМ - этот протокол не используется в мире мультикоптеров. Поддерживается только крайне значения, ограниченные количеством ESC и радиоприемников. Рабочий цикл 0% PWM означает «Стоп», а рабочий цикл 100% PWM означает «Полная мощность». Современные пилоты БПЛА вообще не должны об этом думать. Интересен тот факт, что BLHeli поддерживает его как «PWM Input» параметр, который по умолчанию отключен.

* 1. **STANDARD**

Частота: 50 Гц / 400 Гц

Длительность импульса: 1000 мкс (стоп) – 1860 (2000) мкс (максимальная скорость)

Период: 20000 мкс для 50 Гц / 2500 мкс для 400 Гц

Протокол кодирует запрашиваемый выходной сигнал как длину импульса. Длительность импульса 1 мс означает «Стоп», а длина импульса 2 мс означает «Полная мощность». Из-за этого максимальная теоретическая частота обновления составляет 500 Гц (490 Гц в практических приложениях). Задержка сигнала в случае протокола PWM составляет 2 мс. Это означает, что ESC может начать обновление вывода через 2 мс после того, как полетный контроллер начал отправлять эту информацию. Все это делает ШИМ медленным, и использование времени цикла ниже 2000 мкс (частота обновления 500 Гц) не имеет смысла.

* 1. **ONESHOT125**

Частота: 1 кГц / 2 кГц

Длительность импульса: 125 мкс (стоп) – 230 (250) мкс (максимальная скорость)

Период: 1000 мкс для 1 кГц / 500 мкс для 2 кГц

Этот протокол использует в 8 раз более короткие импульсы, чем стандартный протокол ШИМ: от 125 мкс (остановка) до 250 мкс (полная мощность). Это означает 2 вещи: он обеспечивает в 8 раз более высокую скорость обновления контура ПИД-регулирования (время цикла 250 мкс / частота обновления 4 кГц). Он также имеет в 8 раз меньшую задержку сигнала: всего 250 мкс вместо 2000 мкс. В настоящее время OneShot125 - это минимум для мини-квадрокоптеров. Даже большие машины оценят меньшую задержку. Поддерживается большинством полетных контроллеров и ESC (SimonK, BLHeli, KISS, другие). Если и прошивка Flight Controller, и ESC поддерживают OneShot125, его следует использовать.

* 1. **ONESHOT42**

Частота: 1 кГц / 2 кГц / 4 кГц / 8 кГц

Длительность импульса: 42 мкс (стоп) – 78 мкс (максимальная скорость)

Период: 1000 мкс для 1 кГц / 500 мкс для 2 кГц / 250 мкс для 4 кГц / 125 мкс для 8 кГц

В 3 раза более быстрая версия OneShot125. Максимальная частота обновления 12 кГц и задержка сигнала 42 мкс. Он был разработан Flyduino в рамках «программы» KISS FC и ESC. Пока не получил широкой поддержки.

* 1. **MULTISHOT**

Частота: 1 кГц / 2 кГц / 4 кГц / 8 кГц / 16 кГц / 32 кГц

Длительность импульса: 5 мкс (стоп) – 22 мкс (максимальная скорость)

Период: 1000 мкс для 1 кГц / 500 мкс для 2 кГц / 250 мкс для 4 кГц / 125 мкс для 8 кГц / 62,5 мкс для 16 кГц / 31,25 мкс для 32 кГц

Cамый быстрый протокол ESC в этом сравнении, разработанный RaceFlight, обеспечивает частоту обновления 32 кГц. Он почти в 10 раз быстрее, чем OneShot125 (в 80 раз быстрее, чем PWM). Требуется как быстрый FC (предпочтительно STM32F4), так и быстрый ESC (предпочтительно Silabls F390). Не имеет широкой поддержки, в основном из-за ограниченного количества регуляторов скорости Multishot ready.

* 1. **BRUSHED**

Частота: 8 кГц / 16 кГц / 32 кГц

Длительность импульса:

8 кГц: 0 мкс (стоп) – 106 мкс (максимальная скорость)

16 кГц: 0 мкс (стоп) – 53 мкс (максимальная скорость)

32 кГц: 0 мкс (стоп) – 26 мкс (максимальная скорость)

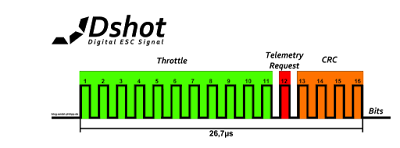
Период: 125 мкс для 8 кГц / 62,5 мкс для 16 кГц / 31,25 мкс для 32 кГц

* 1. **Протоколы DSHOT**

Протоколы ШИМ, такие как Multishot, Oneshot42 и Oneshot125, полагаются на ширину импульса, чтобы указать положение дроссельной заслонки. Это может привести к возникновению таких проблем, как джиттер, когда временные колебания вызывают ошибки в желаемой скорости дроссельной заслонки. Чем короче ширина импульса протокола, тем выше вероятность и влияние джиттера

Преимущество DShot перед OneShot не столько в скорости, сколько в надежности и гибкости. Переход к полностью цифровому протоколу позволяет ввести исправление ошибок и двунаправленный поток данных. Кроме того, с DShot калибровка ESC не требуется. Калибровка заключается в настройке ширины импульса ШИМ, которая не требуется для цифрового протокола. Последним преимуществом DShot является более высокое разрешение значения дроссельной заслонки (2000 шагов), что должно привести к более плавному управлению.

Цифровые ноль и единица протокола DShot определяются длительностью импульса. Пакет данного протокола состоит из 16 бит: 11 бит со значением скорости дроссельной заслонки, 1 бит запроса телеметрии и 4 бита CRC12.



- Значения от 0 до 47 в 11 битном значении скорости зарезервированы, 48 – 2047 управляют заслонкой. 0 – снятие с охраны или стоп мотор. Наиболее значимые биты являются первыми:

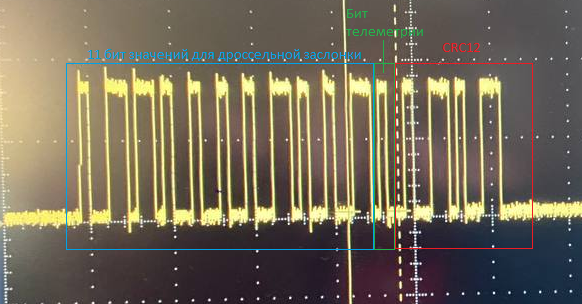
0b111\_1111\_1111 (2047) – полный газ

0b100\_0000\_0000 (1024) – половинный газ

0b11\_0000 (48) – минимальный газ

- Бит запроса телеметрии выставляется, если полетный контроллер запрашивает данные телеметрии у ESC. Сами данные телеметрии от ESC передаются по отдельному проводу.

- CRC12 содержит 4-битное значение и вычисляется по 11 битам данных и биту запроса телеметрии.



Осциллограмма пакета DShot300

На рисунке выше передается сообщение:

Данные заслонки: 0b01100001001 – 777

Бит телеметрии: 0

Crc12: 0b0101 – 5

Сдвигаем значение заслонки на один бит влево и делаем логическое или с битом телеметрии: 777 << 1 | 0 = 1554. Рассчитываем crc12 для полученного значения: 5.

uint8\_t dshot\_get\_crc12 (uint16\_t input\_value)

{

uint16\_t chechsum = 0;

uint16\_t chechsum\_data = input\_value;

for (uint8\_t i = 0; i < 3; i++) {

chechsum ^= chechsum\_data;

chechsum\_data >>= 4;

}

chechsum &= 0xF;

return chechsum;

}

47 зарезервированных команд в 11-битном поле данных скорости:

//взято из src/main/drivers/pwm\_output.h в github’е прошивки betaflight

typedef enum {

DSHOT\_CMD\_MOTOR\_STOP = 0,

DSHOT\_CMD\_BEACON1,

DSHOT\_CMD\_BEACON2,

DSHOT\_CMD\_BEACON3,

DSHOT\_CMD\_BEACON4,

DSHOT\_CMD\_BEACON5,

DSHOT\_CMD\_ESC\_INFO, // V2 includes settings

DSHOT\_CMD\_SPIN\_DIRECTION\_1,

DSHOT\_CMD\_SPIN\_DIRECTION\_2,

DSHOT\_CMD\_3D\_MODE\_OFF,

DSHOT\_CMD\_3D\_MODE\_ON,

DSHOT\_CMD\_SETTINGS\_REQUEST, // Currently not implemented

DSHOT\_CMD\_SAVE\_SETTINGS,

DSHOT\_CMD\_SPIN\_DIRECTION\_NORMAL = 20,

DSHOT\_CMD\_SPIN\_DIRECTION\_REVERSED = 21,

DSHOT\_CMD\_LED0\_ON, // BLHeli32 only

DSHOT\_CMD\_LED1\_ON, // BLHeli32 only

DSHOT\_CMD\_LED2\_ON, // BLHeli32 only

DSHOT\_CMD\_LED3\_ON, // BLHeli32 only

DSHOT\_CMD\_LED0\_OFF, // BLHeli32 only

DSHOT\_CMD\_LED1\_OFF, // BLHeli32 only

DSHOT\_CMD\_LED2\_OFF, // BLHeli32 only

DSHOT\_CMD\_LED3\_OFF, // BLHeli32 only

DSHOT\_CMD\_AUDIO\_STREAM\_MODE\_ON\_OFF = 30, // KISS audio Stream mode on/Off

DSHOT\_CMD\_SILENT\_MODE\_ON\_OFF = 31, // KISS silent Mode on/Off

DSHOT\_CMD\_SIGNAL\_LINE\_TELEMETRY\_DISABLE = 32,

DSHOT\_CMD\_SIGNAL\_LINE\_CONTINUOUS\_ERPM\_TELEMETRY = 33,

DSHOT\_CMD\_MAX = 47

} dshotCommands\_e;

Примечания к специальным командам:

Если вы изменяете какие-либо настройки с помощью команд DShot, вы должны выполнить команду сохранения настроек, чтобы они вступили в силу;

Чтобы изменить направление вращения, установить 3D-режим или сохранить настройки, вы должны включить бит телеметрии в соответствующем командном пакете, и вы должны ввести команду 10 раз подряд, чтобы команда вступила в силу.

Если вы подаете команду 0, а ESC не активирован, он активирует ESC с установленными диапазонами газа. Если ESC активирован, он останавливает моторы. Эту команду следует использовать, если вы хотите, чтобы двигатель остановился. Если вы попытаетесь использовать настройку дроссельной заслонки, чтобы остановить двигатель, он все равно будет медленно вращаться.

Если вы перестанете отправлять команды, ESC снимет с охраны (10 мс или меньше).

Существует четыре типа (или режимов протокола) DShot, различающихся скоростью подключения:

DShot150 - частота обновления 150 кбит/с или 9375 Гц

DShot300 - частота обновления 300 кбит/с или 18,75 кГц

DShot600 - частота обновления 600 кбит/с или 37,5 кГц

DShot1200 - частота обновления 1200 кбит/с или 75 кГц

Максимальные практические частоты обновления ESC (т. е. частоты цикла Flight Controller) существенно ниже, чем теоретические частоты обновления, указанные выше:

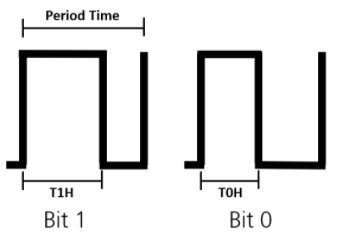
DShot150: 4,05 кГц max.

DShot300: 8,09 - 10,6 кГц max (10,6 кГц доступно только на гироскопических платах 32 кГц)

DShot600: max 16,0 кГц

DShot1200: > 32,0 кГц макс (в настоящее время только прошивка ESC KISS24 поддерживает DSHOT1200)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Режим | Длительность логической единицы, мкс | Длительность логического нуля, мкс | Период, мкс | Пауза между кадрами, мкс |
| Dshot 150 | 5 | 2,5 | 6,68 | 140 |
| Dshot 300 | 2,5 | 1,25 | 3,34 | 70 |
| Dshot 600 | 1,25 | 0,625 | 1,67 | 35 |
| Dshot 1200 | 0,625 | 0,313 | 0,835 | 17,5 |



Длительность импульса для логических единицы и нуля

Длительность логической единицы составляет 74,85% от периода, длительность логического нуля – 37,425%. Пауза (или сброс) между сообщениями (кадрами) должна составлять более 2 мкс (рекомендуют 21 бит, в таблице приведены данные для 21 периода, а не бита). Сброс просто указывает на конец одного кадра, и, таким образом, любые будущие биты являются началом нового кадра. Поскольку DShot происходит в конце цикла PID контроллера полета, эта пауза на самом деле значительно длиннее. Если бы DShot должен был постоянно выводить сигнал, тогда эта задержка потребовалась бы.

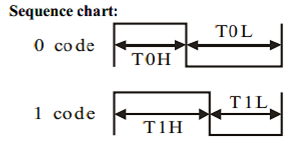


Диаграмма последовательности

Для инициализации ESC нужно проделать то же самое, что и с ШИМ протоколами: слать нулевое значение некоторое время. Но для некоторых ESC для их инициализации придется слать значения скорости от максимального до минимального и опять выждать чтобы услышать «мелодию» готовности.

Для лучшей реализации «в железе» лучше использовать режим прямого доступа к памяти (DMA) для ШИМ режима работы таймера.

Пока ESC не «просигналит» готовность, слать в него какие-то данные не стоит.

При попытке послать сразу большое значение скорости (или большую дельту, от 0 до 1000), ESC (или, возможно, БП) включает защиту по потреблению.

1. **Архитектура программного обеспечения**



Пунктирной линией выделено то, что предстоит разработать.

drone\_(выполняемая функция) – это слой-обертка на библиотечные функции, регистры. Его использование позволит выбирать на разных этапах разработки более подходящую готовую библиотеку от производителя. Как пример: в начале используем HAL как более удобную и позволяющую быстрее разработать код, выполняющий требуемую функциональность, но вместе с тем, более затратную по памяти и процессорному времени; далее для большей оптимизации некоторые функции HAL заменяются на более производительные, но и более сложные в плане понимая низкоуровневых процессов, библиотеку LL и/или регистры.

1. **Полетный контроллер (плата управления). Теория**

Основная задача платы полетного контроллера – это упростить управление дроном и сделать его максимально «устойчивым».

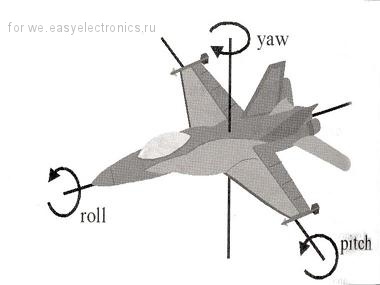
Для эффективного управления системной, нужны 3 вещи:

1. Знать, в каком состоянии хотим поддерживать систему
2. Иметь возможность определять, в каком состоянии находится система в данный момент
3. Иметь возможность оказывать управляющее воздействие на систему

Для определения текущего состояния системы служат датчики: гироскопы, акселерометры, магнитометры и тд. Управляющим воздействием является изменение скорости вращения двигателей / винтов. Например, нужно развернуть коптер вокруг определенной горизонтальной оси – увеличиваем обороты одного двигателя и снижаем обороты противоположенного двигателя. Изменяя, на одинаковое значение, обороты сразу всех двигателей, мы изменяем общую подъемную силу, и можем управлять положением ЛА по высоте.

Что касается пункта 1): есть два подхода – управление по угловой скорости вращения вокруг осей и по абсолютному положению ЛА.

**8.1 Управление по угловой скорости**



Есть три угла: тангажа (pitch), крена (roll) и рысканья (yaw).

Первая проблема, с которой приходится сталкиваться при управлении коптером – он самопроизвольно вращается по всем этим углам. Причем быстро, спонтанно и по всем трем одновременно. Если стабилизировать вращение по данным углам – поведение ЛА будет более предсказуемым, и управлять ним будет на порядок легче.

Алгоритм управление в данном случае такой:

1. Получаем моментальное значение угловой скорости с гироскопа (для данного угла).  
  
2. Нормируем полученное значение, и умножаем на некий коэффициент P. Этот коэффициент – пареметр пропорционального регулирования.  
  
3. Изменяем скорость вращения двигателей (на одной из осей коптера) на величину, полученную на предыдущем этапе. Для одного двигателя уменьшаем скорость вращения (на указанную величину), для противоположенного – увеличиваем. Это создает тягу, направленную в противоположенную сторону от вращения коптера (измененного на этапе 1), и тормозит вращение.  
  
Алгоритм (1 – 3) повторяется для каждого из трех углов. После чего цикл начинается сначала.

При правильно подобранном коэффициенте P регулятор будет достаточно эффективно «тормозить» вращение коптера.

Данный регулятор просто пытается свести к нуля значение угловой скорости, он не знает про положение ЛА относительно горизонта, не берет в расчет линейное перемещение ЛА. Если дрон перевернулся на 180 градусов и отвесно падает – регулятор на это не отреагирует, если нет вращения по осям (есть только линейное ускорение), регулировать нечего.

**8.2 Стабилизация по абсолютному положению ЛА**

Такой подход возлагает на плату управления стабилизацию по всем параметрам: углам, высоте, линейным перемещениям – в идеале, такая плата должна удерживать дрон неподвижно в воздухе и противодействовать любым внешним воздействиям.

Самая большая сложность данного подхода – получение данных об абсолютном положении ЛА в пространстве с достаточной точностью и оперативностью. Для получения данных применятся БИНС, построенный на MEMS приборах: акселерометры, гироскопы, магнетометры и т.д.

1. **Аббревиатуры и термины**

**GCS** – наземная станция управления

**WP** – путевая точка

**RTH** – режим «Вернуться домой». Опция запускается принудительно и возвращает беспилотник на место взлета или включается автоматически при потере сигнала или критично низком заряде аккумулятора.

**PH** – режим «Удержание позиции»

**Headless mode** - Безголовый режим. Коптер ориентируется только на сигнал пульта ДУ, а не на то, куда повернута его голова

**Failsave (FailSafe) Mode** - Безаварийный режим, который срабатывает, когда с коптером потеряна связь. При активации «Failsave Mode» мультикоптер сделает то, что вы определили в настройках: прилетит на указанную точку взлета на определенной высоте и совершит безопасную посадку.

**Hold** - Если радиосигнал потерян – модель выполняет последнюю команду, полученную с передатчика, и так до тех пор, пока пилот не возьмет аппарат под свой контроль. Функция Hold была разработана для того, чтобы авиамодели на радиоуправлении могли благополучно перелетать «мертвые» зоны или зоны повышенной «зашумленности».

**Follow Me** - Следуй за мной. Включите эту функцию, и дрон проследует за вами, обновляя точку взлета в своих координатных данных. Положите пульт в карман или сумку и смело идите, поезжайте на велосипеде или машине – коптер будет преследовать вас и при желании – снимать. В случае активации «Return Home» он прилетит к пилоту, а не на изначальную точку взлета.

**Watch Me** - Смотри на меня – режим персональной камеры в небе. Похоже на «Follow Me»: беспилотник следует за вами и постоянно снимает, удерживая человека с пультом ДУ в фокусе объектива.

**Image tracking** - Отслеживание объекта. Опция похожа на «Watch Me» с той разницей, что в фокусе объектива может быть любой объект.

**Полет по заданным точкам («Curve Cable Cam» или «Waypoint»)** - Доступен в профессиональных и любительских дронах. Укажите на карте в приложении точки, куда стоит слетать и посылайте беспилотник на задание. Очень удобно, чтобы не отвлекаться на управление, а только внимательно следить за камерой.

**FPV** - First Person View, вид от первого лица. Такой аббревиатурой называют одно из направлений радиоуправляемого авиамоделизма. В данном случае осуществляется не только управление авиамоделью по радиоканалу системы радиоуправления, но и приём с модели видео изображения по дополнительному видео-радиоканалу в режиме реального времени. Пилот, управляющий авиамоделью, видит изображение, получаемое с видеокамеры при помощи устройств отображения: мониторов, телевизоров, видео-очков, видео-шлемов.

Разделяют две подгруппы направления: Low Range FPV и Long Range FPV. В первом случае используют стандартный набор для FPV, включающий маломощный передатчик видеосигнала, позволяющий летать в зоне действия стандартного передатчика радиоуправления. Для дальних полетов используют усилители мощности или более мощные передатчики как для управления моделью, так и для передачи с неё видеосигнала.

**MAVLink** – это протокол для организации связи между автономными летательными и транспортными системами (дронами, самолетами, автомобилями).

**Частоты передачи**

Используются частоты 27 МГц, 40 МГц и 2,4 ГГц. Последняя - лучше всего, так как помехозащищенная. Так же используют Wi-Fi (малый радиус действия, используется как вспомогательный способ связи).

**Барометр**

Измеряет атмосферное давление. С увеличением высоты, давление падает. Не работает в плохую погоду (сказываются природные перепады атмосферного давления) и на малых высотах.

**Дроссельная заслонка (Стик газа, Throttle)**

Пилот регулирует обороты двигателя, требуется постоянная корректировка стиком газа, чтобы поддерживать высоту. Если стик газа убрать в минимум , то двигатели остановятся

1. **Код**

Что отправляется в handleFrSkyTelemetry():

----------------------------------------------каждые 125 мс--------------------------------------------------------------

-Высота: sendPitch()

-Крен: sendRoll()

ИЛИ

-Ускорение в g: sendAccel

-Вертикальная скорость для opentx в см/с: sendVario()

-------------------------------------------------------------------------------------------------------------------------------

----------------------------------------------каждые 500 мс--------------------------------------------------------------

-Высота барометра: sendBaro(); //каждые 5 с

-Курс (рыскание): sendHeading()

-------------------------------------------------------------------------------------------------------------------------------

----------------------------------------------каждую 1 с-------------------------------------------------------------------

-Режим полета Температура1 (см. Таблицу 2): sendFlightModeAsTemperature1()

-Статус дроссельной заслонки ИЛИ батарейный размер в об/мин (Если флаг ARMED – первое, иначе – второе): sendThrottleOrBatterySizeAsRpm() (см. раздел Аббревиатуры и термины)

Если фича FEATURE\_VBAT определена (а она определена в файле target.h):

-Напряжение батареи / количество ячеек (разрешение 0,01 В)

-Напряжение батареи / количество ячеек (разрешение 0,1 В)

-Ток

-Заряд батареи

Если USE\_GPS определен (а он определен в файле common.h):

Если датчик GPS присутствует

-Скорость по GPS: sendSpeed()

-Расстояние до дома в метрах: sendHomeDistance()

-Высота: sendGpsAltitude()

-Состояние Tmp2 (см. Таблицу 2): sendSatalliteSignalQualityAsTemperature2()

-Широта и Долгота: sendGPSLatLong()

Иначе

- отправить поддельную широту, которая позволяет отображать заголовок

/\*Заголовок отображается только в OpenTX, если также отправлено ненулевое значение широты / долготы\*/

-------------------------------------------------------------------------------------------------------------------------------

----------------------------------------------каждые 5 с-------------------------------------------------------------------

Время: sendTime()

-------------------------------------------------------------------------------------------------------------------------------

**Приложение**

**Инструкция настройки окружения и сборки iNAV в Windows**

1. Установить подсистему Ubuntu на Windows из магазина приложений

2. При первом запуске, скорее всего возникнет ошибка, что-то вроде "installing this may take a few minutes ..." с 16-м кодом ошибки.

Это решается путем запуска Windows-приложения PowerShell от имени админа и прописывания в его консоли следующей команды: Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux (терминал Ubuntu должен быть выключен)

3. Повторно запустить консоль Ubuntu и подождать некоторое время, код ошибки должен пропасть, надпись вроде "take a few minutes ..." - нет.

4. После задания юзером логина и пароля необходимо поставить инструментарий:

*sudo apt update*

*sudo apt-get install git*

*sudo apt-get install make*

*sudo apt-get install cmake*

*sudo apt-get install ruby*

*sudo apt-get install gcc*

или

*sudo apt update && sudo apt upgrade*

*sudo apt install git make ruby cmake gcc*

5. Нужно поставить ARM компилятор:

*sudo apt-get install gcc-arm-none-eabi binutils-arm-none-eabi gdb-arm-none-eabi openocd*

(первый обязательно, что-то откажется ставиться, вроде openocd)

6. Качаем репозиторий. Предварительно нужно смонтировать диск C:

*cd /mnt/c*

Клонируем:

*git clone https://github.com/iNavFlight/inav.git*

Если ошибка:

*Cloning into 'inav'...*

*error: chmod on /mnt/c/inav/.git/config.lock failed: Operation not permitted*

*fatal: could not set 'core.filemode' to 'false'*

Покинь директорию и введи следующее:

*sudo umount /mnt/c*

*sudo mount -t drvfs C: /mnt/c -o metadata*

и вернись к шагу клонирования

7. Получение Makefile

Перейти в директорию inav и создать директорию build:

*mkdir build*

*cd build*

*cmake ..*

Это приведет к генерации cmake-ом Makefile-а

8. Построение:

Перейти в папку build

*cd build*

и сделать make для таргета (в нашем случае это – MATEKF722)

*make MATEKF722*

Ошибка /bin/sh: 1: Syntax error: "(" unexpected

1. В подсистеме Linux *cd /etc/*
2. Создайте новый файл с *sudo nano wsl.conf*
3. Введите в новый файл следующее:

*[Interop]*

*appendWindowsPath=false*

1. Сохраните файл, удерживая Ctrl и нажав O
2. Нажмите, Enter чтобы подтвердить имя файла wsl.conf.
3. Нажмите Ctrl + X чтобы выйти
4. Желательно перезапустить WSL и Windows
5. *cd build*
6. *cmake ..*
7. *make {TARGET}* должен снова работать

PinOut Matek F722

A00: FREE

A01: FREE

A02: FREE

A03: SERIAL2 UART RX

A04: FREE

A05: SPI1 SCK

A06: SPI1 MISO

A07: SPI1 MOSI

A08: FREE

A09: SERIAL1 UART TX

A10: SERIAL1 UART RX

A11: USB IN

A12: USB OUT

A13: FREE

A14: LED2 OUT

A15: FREE

B00: ADC CH3

B01: FREE

B02: FREE

B03: SPI3 SCK

B04: SPI3 MISO

B05: SPI3 MOSI

B06: I2C1 SCL

B07: I2C1 SDA

B08: FREE

B09: LED1 OUT

B10: SPI\_PREINIT CS

B11: FREE

B12: FREE

B13: FREE

B14: FREE

B15: FREE

C00: ADC CH1

C01: SDCARD CS

C02: MPU CS

C03: MPU EXTI

C04: ADC CH2

C05: FREE

C06: MOTOR1 OUT

C07: MOTOR2 OUT

C08: MOTOR3 OUT

C09: MOTOR4 OUT

C10: SERIAL3 UART TX

C11: SERIAL3 UART RX

C12: FREE

C13: BEEPER OUT

C14: FREE

C15: FREE

D02: FREE

Подключение GPS датчика:

Питание +5В