ПРОТОКОЛ ИНФОРМАЦИОННОГО ВЗАИМОДЕЙСТВИЯ C  
ГИРОСТАБИЛИЗИРОВАННОЙ ПЛАТФОРМОЙ

Содержание

1. [Настройки обмена 3](#bookmark1)
2. [Структура командного пакета 3](#bookmark4)
3. [Перечень команд 4](#bookmark7)
4. [Примеры формирования командных сообщений 6](#bookmark10)
5. [Структура пакета с телеметрией 7](#bookmark13)
6. [Алгоритм расчета контрольной суммы 8](#bookmark16)
7. [Последовательность подачи команд для управления 9](#bookmark19)
8. Настройки обмена

Настоящий протокол устанавливает порядок информационного взаимодействия с гиростабилизированной платформой (ГСП) по интерфейсу RS-422.

***Параметры обмена:***

1. Скорость обмена: 256000 бит/c;
2. Биты данных: 8;
3. Четность: нет;
4. Число стоп-битов: 1;
5. Структура командного пакета

Управление ГСП осуществляется передачей командного сообщения.

Структура командного пакета представлена в таблице 1.

Таблица 1 - Структура командного сообщения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Название | Длина, байт | Смещение, байт | Значение/описание |
| 1 | START BYTE | 1 | 0 | = 0xAA Стартовый байт пакета |
| 2 | CMD | 1 | 1 | Код команды (таблица 2) |
| STATUS | 1 | 2 | Статус-код (таблица 3) |
| SRC | 2 | 3 | = 0x0000 |
| DST | 2 | 5 | Адрес получателя (таблица 4) |
| 3 | WORD\_CNT | 2 | 7 | Длина поля данных (количество 32 битных слов данных) |
| 4 | DATA | WORD ONT \* 4 | 9 | Данные |
| 5 | CRC | 1 | 8 + WORD CNT \* 4 | Контрольная сумма |

Таблица 2 - Возможные значения поля Код команды (CMD) командного пакета

|  |  |
| --- | --- |
| Значение | Описание |
| 0x00 | Запрос на запись |
| 0x01 | Запрос на чтение |
| 0x81 | Ответ на чтение |

Таблица 3 — Возможные значения поля Статус-код (STATUS) командного пакета

|  |  |
| --- | --- |
| Значение | Описание |
| 0x00 | Запрос без ответа |
| 0x80 | Запрос с ответом/Успешный ответ |

1. Перечень команд

Отправка 2х слов данных возможна только для управления по скорости, см. пример в п.4.1

Таблица 4 — Описание регистров

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Адрес | Доступ на запись/чтение (R/W) | Название | Тип данных | Назначение | Диапазон значений |
| 0x00FE | - | Telemetry | (Таблица 5) | Отправляемая изделием телеметрия |  |
| 0x00F1 | W | CPUReset | uint32\_t | Сброс CPU | Сброс с последующим перезапуском и переинициализацией CPU. Последовательность запуска изделия, как после сброса питания. |
| 0x0001 | W | MotorControl | uint32\_t | Включение двигателей: 1 – вкл, 0 – выкл. | 0 | 1 |
| 0x0002 | W | StabControl | uint32\_t | Включение стабилизации: 1 – вкл, 0 – выкл. | 0 | 1 |
| 0x0010 | W | StabSpeedZ | float | Управление по скорости по азимуту, [град/сек] | ± 60.0° |
| 0x0011 | W | StabSpeedX | float | Управление по скорости по углу места, [град/сек] | ± 60.0° |
| 0x0012 | W | StabAAngleZ | float | Управление по углу места, связанному с платформой, [град] | ± 180.0° |
| 0x0013 | W | StabAAngleX | float | Управление по углу азимута, связанному с платформой, [град] | ± 60.0° |
| 0x0014 | W | StabRAngleZ | float | Управление по углу места, относительно текущего угла, [рад] | ± 180.0° |
| 0x0015 | W | StabRAngleX | float | Управление по углу азимута, относительно текущего угла, [рад] | ± 180.0° |
| 0x0020 | W | MemsCalibration | uint32\_t | Автоматическая калибровка дрейфа гироскопа. Во время калибровки устройство не должно подвергаться внешним воздействиям и вибрациям. Если после калибровки, в режиме стабилизация, присутствует дрейф, калибровку следует повторить. Полученные результаты калибровки можно сохранить командой в энергонезависимую память. | 1 мс .. 15000 мс  0 - сбросить |
| 0x42 | W | SetAngleOffsetZ | uint32\_t | Установит текущее положение оси Z как нулевое | - 0xFFFFEEEE – установить |
| 0x43 | W | SetAngleOffsetX | uint32\_t | Установит текущее положение оси X как нулевое | - 0xFFFFEEEE – установить |
| 0x0056 | W | SaveAll | uint32 | После настройки одного или нескольких параметров изделия необходимо сохранить их в память устройства, после сохранения параметров требуется перезагрузка устройства. | 0xAAAAFFFF – сброс сохранённых параметров, на параметры по умолчанию |
| 0x00000000 – сохранить параметры |
|  |  |  |  |  | 0x00000000 – сохранить параметры |

1. Примеры формирования командных сообщений
2. *Команда на управление по скорости:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Название | Длина, байт | Смещение, байт | Значение |
| 1 | START BYTE | 1 | 0 | = 0xAA |
| 2 | CMD | 1 | 1 | = 0x00 (таблица 2) |
| STATUS | 1 | 2 | = 0x00 (таблица 3) |
| SRC | 2 | 3 | = 0x0000 |
| DST | 2 | 5 | = 0x0010 (таблица 4) |
| 3 | WORD CNT | 2 | 7 | = 0x0002 |
| 4 | DATA1 | 4 | 9 | = 5.0 ° (скорость по азимуту) |
| 5 | DATA2 | 4 | 13 | = -1.0 ° (скорость по углу места) |
| 6 | CRC | 1 | 17 | = 0xXX |

1. Структура пакета с телеметрией

Отправка телеметрии начинается автоматически с частотой 30 Гц. Структура пакета телеметрии соответствует таблице 5.

Таблица 5 — Структура пакета телеметрии

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Название | Длина, байт | Тип данных | Описание | Значение |
| 1 | START BYTE | 1 | uin8 t |  | = 0xAA |
| 2 | CMD | 1 | uin8 t |  | = 0x81 (таблица 2) |
| STATUS | 1 | uin8 t |  | = 0x00 (таблица 3) |
| SRC | 2 | uin16 t |  | = 0x00FE |
| DST | 2 | uin16 t |  | = 0x0000 |
| 3 | WORD CNT | 2 | uin16 t |  | = 0x000A |
| 4 | State | 4 | uin32 t | Состояние изделия (таблица 6) |  |
| 5 | Errors | 4 | uin32 t | Ошибки изделия (таблица 7) |  |
| 6 | AngleZ | 4 | float | Значение угла по азимутy, [градус] | = 360.0 ° |
| 7 | AngleX | 4 | float | Значение угла места, [градус] | = 360.0 ° |
| 8 | SpeedZ | 4 | float | Скорость по азимуту, [градус/сек] | = ± 60.0 ° |
| 9 | SpeedZ | 4 | float | Скорость по углу места, [градус/сек] | = ± 60.0 ° |
| 10 | MemsSpeedX | 4 | float | Значение скорости гироскопа оси X, [градус/сек] |  |
| 11 | MemsSpeedY | 4 | float | Значение скорости гироскопа оси Y, [градус/сек] |  |
| 12 | MemsSpeedZ | 4 | float | Значение скорости гироскопа оси Z, [градус/сек] |  |
| 13 | Reserve | 4 | uin32 t | Резерв |  |
| 14 | CRC | 1 | uin8 t | Контрольная сумма |  |

Данные в пакете телеметрии упаковываются в порядке «сначала младший — потом старший».

Таблица 6 — Значения битов состояния

|  |  |  |
| --- | --- | --- |
| Бит | Обозначение | Описание |
| 0 | Ready | = 1 - изделие корректно инициализировано и готово к работе  *= 0 -* изделие не инициализировано и к работе не готово |
| 1 | Stabilization | = 1 - стабилизация включена;  = 0 - стабилизация выключена; |
| 2 | Motors | = 1 - моторы включены;  = 0 - моторы выключены. |
| 3 | SelfTesting | = 1 - запущенно самотестирование; |
| 4 | Calibration | = 1 - запущенна калибровка; |
| 5 | NotSaved | = 1 – есть не сохранённые данные; |
| 7..31 |  | резерв. |

Таблица 7 — Значения битов ошибки

|  |  |  |
| --- | --- | --- |
| Бит | Обозначение | Описание |
| 0 | msgCheckingTXRS422 | Наличие сбоев обмена по линии TX RS422. 1 = сбой; 0 – отсутствует. |
| 1 | msgCheckingRXRS422 | Наличие сбоев обмена по линии RX RS422. 1 = сбой; 0 – отсутствует. |
| 2 | msgCheckingCAN | Наличие сбоев обмена по линии CAN. 1 = сбой; 0 – отсутствует. |
| 3 | angleSensorZ | Наличие сбоев датчика угла оси Z. 1 = сбой; 0 – отсутствует. |
| 4 | angleSensorX | Наличие сбоев датчика угла оси X. 1 = сбой; 0 – отсутствует. |
| 5 | mems | Наличие сбоев MEMS. 1 = сбой; 0 – отсутствует. |
| 6 | motorZ | Наличие сбоев двигателя оси Z. 1 = сбой; 0 – отсутствует. |
| 7 | motorX | Наличие сбоев двигателя оси Z. 1 = сбой; 0 – отсутствует. |
| 8 | initMEMS | Ошибка инициализации MEMS. 1 = сбой; 0 – ошибка отсутствует. |
| 9 | initAngleSensorZ | Ошибка инициализации датчика угла оси Z. 1 = сбой; 0 – отсутствует. |
| 10 | initAngleSensorX | Ошибка инициализации датчика угла оси Z. 1 = сбой; 0 – отсутствует. |
| 11 | initMotorZ | Ошибка инициализации мотора оси Z. 1 = сбой; 0 – отсутствует. |
| 12 | initMotorX | Ошибка инициализации мотора оси X. 1 = сбой; 0 – отсутствует. |
| 13..31 |  | резерв. |

1. Алгоритм расчета контрольной суммы

Расчёт значения CRC8 производится для всех байт пакета, включая стартовый, на базе полинома Х8 + Х5 + Х4 + 1 (начальное значение 0xFF), XorOut = 0x00.

#define CRC\_TABLE\_SIZE 256

/\*\* CRC polynome table \*/

uint8\_t \_CRC8Table[CRC\_TABLE\_SIZE];

/\*\* Polynome width \*/ uint32\_t \_poly\_width = 8;

/\*\*

* @brief init\_crc\_calculation
* @param poly - полином
* @param init\_value - начальное значение
* @param xor\_value - значение XorOut
* / void **init\_crc\_calculation**(uint8\_t poly, uint8\_t init\_value, uint8\_t xor\_value) {

\_init\_value = init\_value;

\_xor\_value = xor\_value;

// fill table

const uint32\_t bits\_mask = (1 << \_poly\_width) - 1;

const uint32\_t top\_bit = 1 << (\_poly\_width - 1);

uint32\_t index;

for (index = 0; index < CRC\_TABLE\_SIZE; ++index) {

uint32\_t value = index << (\_poly\_width - 8);

uint32\_t bit\_index;

for (bit\_index = 0; bit\_index < 8; ++bit\_index) {

if (value & top\_bit) {

value = (value << 1) л poly;

} else {

value = value << 1;

}

value &= bits\_mask;

}

\_CRC8Table[index] = value;

}

\_is\_inited = 1;

}

/\*\*

* @brief crc\_calc
* @param data - данные
* @param size - размер данных
* @return
* /

uint8\_t **crc\_calc**(uint8\_t \*data, uint8\_t size) {

uint8\_t crc = \_init\_value;

while (size--) {

crc = \_CRC8Table[crc л \*data++];

}

return crc;

Для инициализации расчета CRC необходимо вызвать функцию инициализации со следующими аргументами: init\_crc\_calculation(0x31, 0xFF, 0x00);