Описание протокола обмена с платой анализатора ЭДА

Ревизия 0.5

Предлагается подход, в котором будет раздельно и независимо друг от друга реализован сетевой протокол для передачи пакетов данных и протокол кодирования данных.

1. Сетевой протокол

На данный момент актуальны три технологии передачи данных с ГТС (генератор тестового сигнала): Zigbee, RS-232, RS-422/485.

Длина одного пакета ограничивается величиной в 241 байт при передаче по проводному каналу связи и 78 байт для передачи по Zigbee.

Примечание: величина полезной нагрузки одного пакета в Zigbee ограничивается 82 байтами, но каждая пересылка пакета между узлами отнимает еще 2 байта, так как планируется использовать передачу данных через промежуточные узлы в случае 104 ГТС (больше 32 ГТС), то имеет смысл заложить возможность передачи тремя скачками. Потому как стандартом не регламентируется поведение при превышении размера полезной нагрузки во время скачка.

Так как размер лога значительно превышает размер полезной нагрузки Zigbee, то предлагается деление данных на пакеты.

Состав пакета при передаче по Zigbee:

1. Количество пакетов (1 байт, беззнаковое целое);
2. Номер пакета в передаче (1 байт, беззнаковое целое);
3. Количество байт данных (1 байт, беззнаковое целое);
4. Данные (75 байт);
5. Контрольная сумма по всему пакету (2 байта);

Состав пакета при передаче по RS-422/485:

1. Адрес конечного устройства (1 байт, беззнаковое целое). Адрес можно передавать с установленным девятым битом, чтобы аппаратно обрабатывать получение сообщения нужным ГТС.
2. Количество пакетов (1 байт, беззнаковое целое);
3. Номер пакета в передаче (1 байт, беззнаковое целое);
4. Количество байт данных (1 байт, беззнаковое целое);
5. Данные (235 байт);
6. Контрольная сумма по всему пакету (2 байта);

Примечание: так как каждый бит передаваемых данных имеет критическое значение, для повышения надежности передачи данных предлагается использовать контрольную сумму **CRC16 с полиномом 0xAC9A (начальное значение 0x0, конечный XOR 0x0)**, который обеспечивает наибольшую надежность для пакетов размером до 241 байта. При проводной передаче данных можно увеличить длину пакета до 256 с использованием другого полинома для расчета контрольной суммы.

Вопросы, которые требуется решить:

* 1. Сделать выбор между RS-422 и RS-485;
  2. Скорость передачи данных по RS-422/485;
  3. Механизм запроса повторной передачи пакета в случае несовпадения контрольной суммы;
  4. Оптимальный размер пакета данных при передаче по проводному каналу;
  5. Отсчет номеров пакета в посылке с 0 или с 1?

1. Протокол данных

Для кодирования данных предлагается использовать широко распространенный подход TLV (Type-length-value). Это двоичная структура с полями «тип», «длина» и «данные». Длина полей «тип» и «длина» фиксированная. Данный подход позволит работать с генераторами разных модификаций и версий протокола, а также значительно упростит синтаксический анализ получаемых бинарных данных.

Так как TLV позволяет вложенность структур, то для поля «тип» достаточно одного октета.

Так как длина данных для отклика может превышать 256 символов, то поле длина состоит из двух октетов.

Предварительный список типов:

- Команда ГТС (получить данные, начать тест и так далее). Если команда без параметров, длина равна 0:

- Запрос текущих данных 0x01

- Установка значений в ГТС

- Текущие данные (тип+4 байта данные)

- Лог теста (содержит служебную информацию и отклики)

- Отклик пауза (4 байта значение тока + 4\*n значения напряжений)

- Отклик разряд (аналогично)

Вопросы, которые требуется решить:

1. Порядок байт (endianness) удобный для работы в микроконтроллере, чтобы не нужно было постоянно вызывать ntohl(), htonl(), ntohs(), htons().
2. Если не установлен датчик внешней температуры или датчик тока, не передавать эти данные или передавать qNaN (см. Приложение 1).
3. Создать общий список типов и разместить, например, в приватном git-репозитории. Определиться с именованием типов (см. Приложение 2).

Сравнение подходов:

На стороне имеющегося протокола простая читаемость человеком получаемых данных.

На стороне предлагаемого подхода:

1. Контроль целостности передаваемых данных
2. Меньшее время передачи данных за счет меньшего объема (приблизительно 8000 байт против 17000 байт в используемом протоколе для тестового лога).
3. Меньшее количество передаваемых пакетов (~110 против 2030 для лога и 1 против 7 для текущих данных), что существенно снижает нагрузку на сеть при передаче по каналу Zigbee.
4. Большая гибкость в версионности используемого протокола.

Приложение 1. Пример запроса текущих данных

Запрос на получение тестовых данных:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0x01 | 0x01 | 0x01 | 0x00 | 0x72 | 0xB5 |
| Всего 1 пакет | Первый пакет | Запрос текущих данных | Длина 0 (Без параметров) | CRC | CRC |

Ответ c ГТС, на котором установлены и внешний датчик температуры, и датчик тока:

|  |  |
| --- | --- |
| 0x01 | Один пакет |
| 0x01 | Первый пакет |
| 0x81 | Ответ на запрос текущих данных |
| 0x18 | Длина текущих данных |
| 0x90 | Тип - Напряжение на аккумуляторе |
| 0x04 | Длина значения |
| 0x45 | Значение напряжения = 2246 мВ |
| 0x0C |
| 0x60 |
| 0x00 |
| 0x91 | Тип – температура на аккумуляторе |
| 0x04 | Длина значения |
| 0x41 | Значение температуры = 20.9 градусов C |
| 0xA7 |
| 0x33 |
| 0x33 |
| 0xA1 | Тип – температура с внешнего датчика |
| 0x04 | Длина значения |
| 0x41 | Значение температуры = 18.6 градусов |
| 0x94 |
| 0xCC |
| 0xCD |
| 0xA2 | Тип – значение датчика тока |
| 0x04 | Длина значения |
| 0xC1 | Значение тока = -23 A |
| 0xB8 |
| 0x00 |
| 0x00 |
| 0xA6 | CRC |
| 0x77 | CRC |

Примечание: в случае, если какой-то датчик не установлен, данные не передаются. Либо передается тихий NaN (qNaN) в соответствии с IEEE 754.

Приложение 2. Пример списка используемых типов для TLV

#define CURRENT\_DATA\_REQUEST\_TYPE 0x01

#define CURRENT\_DATA\_RESPONSE\_TYPE 0x81

#define ACCUMULATOR\_VOLTAGE\_TYPE 0x90

#define ACCUMULATOR\_TEMPERATURE\_TYPE 0x91

#define AMBIENT\_TEMPERATURE\_TYPE 0xA1

#define ACCUMULATOR\_CURRENT\_TYPE 0xA2

Протокол ревизии 0.4

Характеристики UART:

Скорость обмена: 9600 бод

Количество битов данных: 8

Количество стоп битов: 1

Проверка четности: без проверки

Управление потоком: без управления

Символы Команд закодированы в ASCII, после текста команды должены стоять символы возврата каретки и переноса строки “**\r\n**”. Ответ также кодируется в ASCII и оканчивается “**\r\n**”.

Команды (в первой строке дублирующие команды для простоты тестирования):

**1, 2, 3, ixxx, fxxx.x**

**GETUTI, GETU, GETUT, STARTTEST, STARTTEST I=xxx F=xxx.x, GETTESTDATAFAST, GETTESTDATASLOW, SETCURRENT xxx, SETFREQUENCY xxx.x, GETVERSION**

Новые команды (18.02.2016) STARTSLEEP x

**Новые команды (02.03.2016) WRITEDATA и GETDATA**

1. Запрос текущих напряжения и температуры.

При запросе совершается измерение напряжения и температуры. При этом время ответа составляет ... мс. Также есть команда для измерения только напряжения. Напряжение выдается в мВ с точностью до 0,1мВ. Температура в градусах цельсия с точностью до 0,1градусаеуые.

Команды:

**GETUTI, 1**

Ответ в формате:

**START GETUTI**

**U = %f // напряжение на момент проведения теста**

**T = %f // температура на момент теста**

**O = %f // температура второго термодатчика**

**Il = %f // маленький ток**

**Ih = %f // большой ток**

**FINISH GETUTI**

Команда:

**GETU**

Ответ в формате:

**U = %7.1f**

Команда:

**GETUT**

Ответ в формате:

**U = %7.1f; T = %7.1f**

1. Генерация импульсов, измерение и запись.

Команда, запускающая тест со значениями, установленными заранее через **SETCURRENT, SETFREQUENCY**.

Команды:

**STARTTEST, 2**

Ответ в формате:

**START TEST**

**current = %d**

**frequency = %f**

По окончании цикла измерения и записи:

**FINISH TEST**

Команда, запускающая тест со значениями, передаваемыми в теле команды

Команда:

**STARTTEST I=xxx F=xxx.x**

Ответ в формате:

**START TEST**

**current = %d**

**frequency = %f**

По окончании цикла измерения и записи:

**FINISH TEST**

1. Запрос на выдачу лога.

Передача лога с задержкой (**GETTESTDATASLOW**), без задержки (**GETTESTDATAFAST**) или (**3**). Все значения между **START DATA** и **START LOG** пишутся на момент проведения теста.

Команды:

**GETTESTDATASLOW, 3**

**GETTESTDATAFAST**

Ответ в формате:

**START DATA**

**version = %d // версия платы см. GETVERSION**

**current = %d**

**frequency = %f**

**number of samples = %u // кол-во измерений в полупериоде**

**U = %f // напряжение на момент проведения //теста, в мВ (единицу измерения не //ставить)**

**T = %f // температура на момент теста**

**O = %f // температура второго термодатчика**

**Il = %f // маленький ток**

**Ih = %f // большой ток**

**START LOG**

**I = %d \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* %04d**

**//** здесь, первое значение – ток, второе – номер полупериода

**%7.1f // напряжение в мВ, n штук, n=number of samples**

**// Всего (как происходит расчет количества периодов?)**

**FINISH LOG**

**FINISH DATA**

* Если нет каких-то данных, то вместо значения надо подставить символ «-».
* Теперь ток будет передаваться перед значениями полупериода, а не после как раньше.

1. Установка тока.

Задаёт значение тока. Здесь х – десятичные цифры (не менее одной и не более трёх).

Команда:

**SETCURRENT xxx**

Ответ в формате:

**current = %s // выводится считанная строка**

1. Установка частоты.

Задаёт значение частоты. Здесь х – десятичные цифры (не менее одной и не более четырёх).

Команда:

**SETFREQUENCY xxx.x**

Ответ в формате:

**frequency = %s // выводится считанная строка**

1. Запрос ревизии платы.

Выдает версию платы. Привязку нумерации можно выбрать любую, нам главное, чтобы можно было отличить.

Команда:

**GETVERSION**

Ответ в формате:

**version = %d**

1. Ответ в случае неверной команды. (**УБРАТЬ, ответа нет**)

Ответ в формате:

WRONG COMMAND %s // выводится неверная строка

1. Команда уснуть.

По этой команде плата переходит в режим пониженного энергопотребления

Команда уснуть:

**SLEEP**

Ответ в формате:

**SLEEPWG**

Команда проснутся:

**WAKEUP**

Ответ в формате:

**AWAKENED**

1. Команда записать в прибор данные.

Запись от 1 до 7 параметров в память прибора, параметры могут передаваться не все, те которые не передаем, не переписываются.

Команда:

**WRITEDATA QA=double QB=double UC=double UD=double TE=double TF=double PN=string**

**Или**

**WRITEDATA UC=double UD=double**

Ответ в формате:

**WRITE SUCCESSFUL**

1. Команда прочитать данные из прибора.

Передача всех записанных параметров

Команда:

**GETDATA**

Ответ в формате:

**QA = double**

**QB = double**

**UC = double**

**UD = double**

**TE = double**

**TF = double**

**PN = string**

**FINISH GETDATA**