Описание задачи

Имеется некое устройство для чтения бесконтактных карт (ридер с экранчиком). Устройство может подключаться через несколько различных физических портов (Serial, Ethernet/UDP, USB и т. д.) и использует задокументированный низкоуровневый бинарный протокол обмена (см. Приложение 1) поверх которого посылки кодируются в JSON. Устройство состоит из 6 независимых подсистем, каждая из которых обрабатывает входящие сообщения синхронно.

Требуется реализовать некую библиотеку классов на языке C/C++ (стандарт 11-17) для работы с этим устройством. Назначение библиотеки — абстрагироваться от физического способа подключения и низкоуровневого бинарного протокола обмена. Реализацию передачи бинарных сообщений через каждый конкретный физический порт делать не нужно — это задача других модулей.

Решение задачи в «режиме, приближенном к реальности», т.е. если информации в постановке задачи недостаточно, можно задавать вопросы. Можно предложить некое решение, обсудить, потом сделать. Можно сделать сразу.

Примеры посылок:
Исходящая:
> 4143 # HEADER: two ASCII bytes ’AC’
> 002B # LENGTH: 16-bit number, encoded length of 43 bytes
> 01 # MSGID: message id 01h
> 04 # MODULE: module number 04h
> 01 # TYPE: *Command* message
> 7b0a224c656473223a7b0a22626c7565223a310a22726564223a300a22677265656e223a300a7d7d # PAYLOAD: {\n”Leds”:{\n”blue”:1\n”red”:0\n”green”:0\n}}
> A545 # CHECKSUM: 16-bit number of XMODEM CRC16

Successful response:
< 4143 # HEADER: two ASCII bytes ’AC’
< 0003 # LENGTH: 16-bit number, encoded length of 3 bytes
< 01 # MSGID: message id 01h, it must match *Command* message
< 01 # MODULE: module number 01h
< 02 # TYPE: *Successful response* message
< # PAYLOAD: absent because this *Command* message does not imply any response data
< 7EE2 # CHECKSUM: 16-bit number of XMODEM CRC16

Failure response:
< 4143 # HEADER: two ASCII bytes ’AC’
< 002C # LENGTH: 16-bit number, encoded length of 44 bytes
< 01 # MSGID: message number 01h, it must match *Command* message
< 04 # MODULE: module number 04h (Service operations module)
< 03 # TYPE: *Failure response* message
< 7b0a226572726f72436f6465223a310a2274657874223a22444556494345204953204255359220a7d # PAYLOAD: {\n”errorCode”:1\n”text”:”DEVICE IS BUSY”\n}
< 0518 # CHECKSUM: 16-bit number of XMODEM CRC16

Приложение 1. Описание протокола обмена.

Message overview

2.1 Description

Message is bytes sequence in binary format, that consists of the following fields:
| HEADER | LENGTH | MSGID | MODULE | TYPE | PAYLOAD | CHECKSUM |

2.2 Size

Size of the message is variable, but at least 9 bytes for messages without *PAYLOAD* field and up to 4230 bytes.

2.3 HEADER

2.3.1 Description

This field allows easily find start of the message in the byte sequence.
Encoded as two ASCII chars: ‘A’ and ‘C’ (4143 in hex).

2.3.2 Size

2 bytes.

2.4 LENGTH

2.4.1 Description

Defines the length of the included fields, except *HEADER*, *LENGTH* itself and *CHECKSUM*. Encoded as 16-bit BigEndian number.
Includes lengths of following fields:

- MSGID
- MODULE
- TYPE
- PAYLOAD

2.4.2 Size

2 bytes.

2.5 MSGID

2.5.1 Description

Continuously incremented message id, starts from 0 and wraps to 0 after 255. This field intended to be synchronization mark between *IR* and *Host*, that prevents processing messages out of order.

2.5.2 Size

1 byte.

2.6 MODULE

2.6.1 Description

This field describes *module* within *IR* software. It’s meaning depends from the source of the message:

- if message issued by *IR*, then field describes *module* that issued this message
- if message issued by *Host*, then field describes *module* for which the message is intended

Possible values:

- **01h**: Contact cards module
- **02h**: Contactless Level 1 module
- **03h**: Mifare cards module
- **04h**: Service operations module
- **05h**: NXP NTAG cards module
- **06h**: Graphic User Interface (GUI) module

Depending of loaded firmware some *module*s may be absent (unsupported).

2.6.2 Size

1 byte.

2.7 TYPE

2.7.1 Description

This field describes *type* of message. This field was added to simplify message processing allowing perform some intermediate processing at transport (instead of application) layer of protocol handling.

Ex. to reset timer used for response timeout when *Pending* response is arrived. Or reroute *Notification* message to logging entity within *Host* application while waiting for final response to *Command* message.
Possible values:

- **01h**: Command message
- **02h**: Successful (final) response message
- **03h**: Failure (final) response message
- **04h**: Pending (intermediate) response message

2.7.2 Size

1 byte.

2.8 PAYLOAD

2.8.1 Description

This field contains application payload, that depends from *MODULE* and *TYPE* fields of current message.
Content of this field described by *JSON* files and encoded by their rules.
This field can be empty or contains single *message*.
It worth mentioning that “decoder” of payload *must* know what is encoded in it. To distinct between various payloads that may be included in message’s payload, One Of feature is used.

2.8.2 Size

Variable (including 0), depends of *message* that used for encoding.

2.9 CHECKSUM

2.9.1 Description

This field used to check integrity of the message.
This field encoded as XMODEM’s CRC16 stored as 16-bit BigEndian number.
Calculated by whole message excluding itself.

2.9.2 Size

2 bytes.