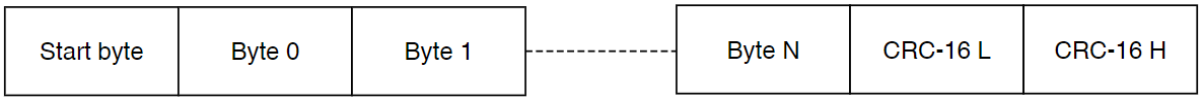
Sheller

Библиотека *Sheller* предназначена для организации пакетного обмена данными между устройствами с гарантией целостности данных.

Передаваемый пакет состоит из начального байта (Start-byte), который служит меткой для начала идентификации пакета, и байтами контрольной суммы CRC-16 (CRC-byte) для определения целостности пакета. Структура пакета:



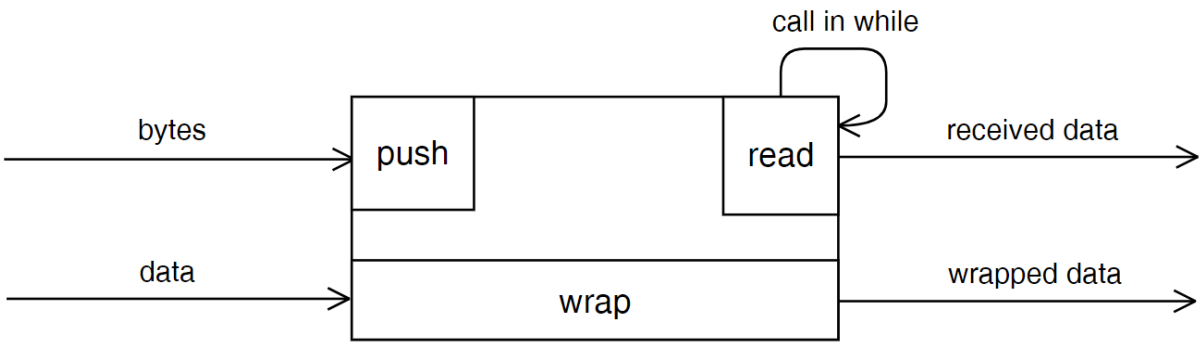
Библиотека использует алгоритм подсчета контрольной суммы CRC-16 – количество байт контрольной суммы равняется 2, а максимальное количество информационных байт 4095. Предустановленное количество информационных байт 8. Имеем: 8 байт полезной информации и 3 сервисных байта. Для ускорения процесса вычисления контрольной суммы используется табличный метод. Размер таблицы 512 байт.

Инициализация объекта *Sheller`a* выполняется функцией:

```
uint8_t sheller_init(sheller_t *desc, uint8_t start_byte, uint8_t usefull_data_length, uint16_t rx_buff_length);
```

В функцию передается указатель на объект sheller`a, значение стартового байта, максимальное количество информации в пакете, размер буфера приема.

Взаимодействие с библиотекой происходит с помощью 3х функций: wrap, push, read:



Для формирования пакета следует передать в функцию wrap указатель на объект sheller`a, указатель на данные для отправки, длину данных и указатель на буфер, в который будет записан пакет.

```
uint8_t sheller_wrap(sheller_t *desc, uint8_t *data, const uint8_t data_length, uint8_t *dest);
```

Длинна данных для отправки не должна превышать значение *SHELLER_USEFULL_DATA_LENGTH*, иначе функция wrap вернет false и не выполнит формирование пакета.

Размер буфера для хранения сформированного пакета для отправки рассчитывается следующим образом: *SHELLER_PACKAGE_LENGTH*. По умолчанию это *8 + 3 = 11.* Сформированный пакет сразу готов к отправке по каналу связи после вызова *wrap*. Размер пакета не зависит от длины данных пользователя и имеет константную длину *SHELLER_PACKAGE_LENGTH.*

На стороне получателя пакет принимается побайтно. Принятые байты заносятся в Sheller с помощью функции *push*.

```
uint8_t sheller_push(sheller_t *desc, const uint8_t byte);
```

В функцию передается указатель на объект Sheller`a и принятый байт по каналу связи. Функция возвращает результат занесения байта во внутренний кольцевой буфер. При переполнении буфера функция возвращает false.

В бесконечном цикле программы следует как можно чаще вызывать неблокирующую функцию *read*.

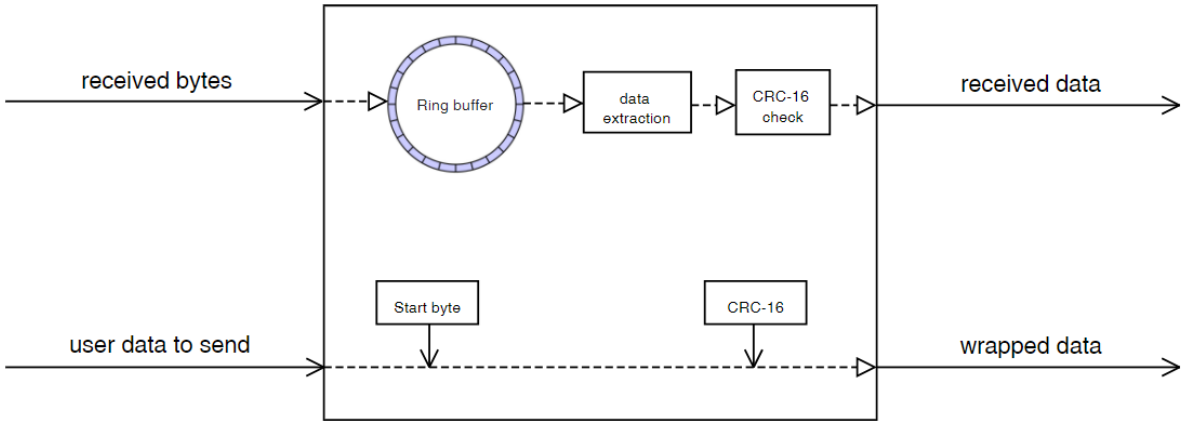
```
uint8_t sheller_read(sheller_t *desc, uint8_t *dest);
```

В функцию передается указатель на объект Sheller`a и указатель на буфер, в который будет записан принятый пакет. Вызов функции *read* запускает процесс выделения пакета данных из циклического буфера. Функция возвращает *true* в случае успешного чтения пакета, в противном случае – false.

Размер циклического внутреннего буфера указан макросом *SHELLER_RX_BUFF_LENGTH.*

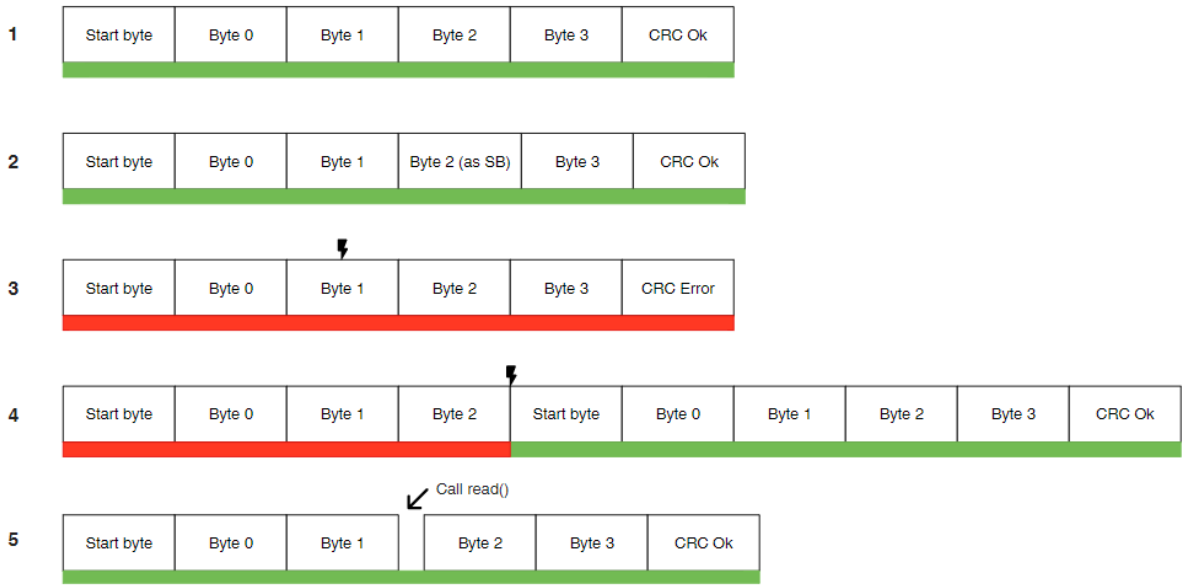
*Переполнение буфера функцией *push* возможно в случае низкой частоты вызова функции *read* или слишком высокой частоты добавления принятых байт и маленького размера буфера *SHELLER_RX_BUFF_LENGTH*.

Логика работы с Sheller`ом:



Sheller предполагает работу в каналах связи с высоким влиянием помех. Функционирование в таких условиях достигнуто машиной состояний (State-machine), которая используется для побайтного приема данных и алгоритмом контрольной суммы.

На иллюстрации ниже приведены примеры влияния помех на передаваемые пакеты:



В 1 случае пакет дошел до получателя без повреждений.

Случай 2 демонстрирует, что Sheller не накладывает ограничения на передаваемые данные. В данных пользователя могут встречаться сервисные байты.

В 3 случае помеха изменила значение второго байта. В результате на принимающей стороне не сошлась контрольная сумма и пакет не был передан в бизнес-логику.

В 4 случае из-за помехи или отсутствии контакта была утеряна часть пакета. На принимающей стороне не сошлась контрольная сумма. В таком случае был проведен поиск следующего начального байта, после чего сходство контрольной суммы и передача пакета в бизнес-логику приложения.

В тесте 5 показано, что функция чтения может быть вызвана над кольцевым буфером, даже если пакет принят частично.