РЕАЛИЗАЦИЯ ПРОТОКОЛА ПЕРЕДАЧИ ДАННЫХ В БОРТОВОЙ СЕТИ СВЕРХМАЛОГО КОСМИЧЕСКОГО АППАРАТА

В.А. Прохоров, М.А. Павлышко, А.В. Никифоров, А.М. Огурцов, А.А. Шандицев

Белорусский государственный университет, Минск, Беларусь
E-mail: prokhorov@bsu.by

На основе решений для бортовой системы управления научно-образовательного наноспутника БГУ BSUSat-2, успешно функционирующего на околоземной орбите, разработаны библиотека и протокол интерфейса *CAN* для взаимодействия модулей созданной инженерной модели сверхмалого космического аппарата (СМКА), позволяющие выполнять отладку бортовых систем на этапе ввода в эксплуатацию и проводить исследовательские эксперименты в учебном процессе при подготовке специалистов аэрокосмической отрасли. Средства разработки: CMake, C, Python.

Ключевые слова: СМКА; BSUSAT-2; CAN.

Ввиду сложности передачи больших объемов данных по интерфейсу *CAN* и для обеспечения модульности и совместимости всех бортовых систем СМКА, в ходе разработки BSUSat-2 были созданы специальные библиотека и протокол для взаимодействия всех бортовых систем СМКА посредством *CAN*–шины [1,2].

Протокол взаимодействия решает сразу несколько задач: передачу больших объемов информации, унификацию взаимодействия устройств на шине, рассылку широковещательных сообщений, работу с конкретным устройством, одновременную асинхронную обработку сообщений, в том числе, от одного отправителя. Библиотека, обеспечивающая поддержку протокола, была разработана на языке *C* и может использоваться на любом устройстве, поддерживающем язык *C*.

Основная идея заключается в разбиении сообщения на части, которые передаются в шину *CAN*, после чего устройство, которому адресовано сообщение, получает *CAN*–кадры и восстанавливает из них исходное сообщение. На рис. 1 представлен заголовок протокола передачи данных.

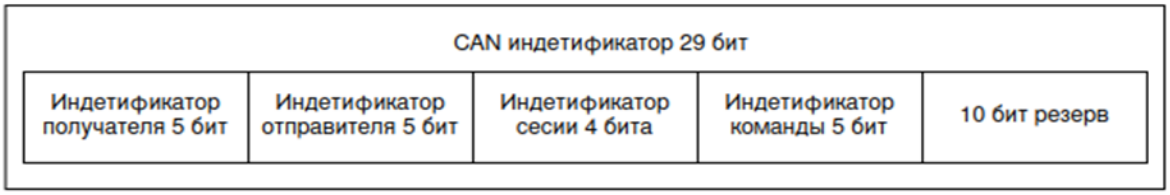

Рис. 1. Заголовок протокола передачи данных

В системе зарезервированы следующие команды для пиринговой передачи: начало передачи (START), передача данных (DATA), конец передачи (END), остановка (ABORT), подтверждение приема команды (ACK), отклонение команды (NACK).

Для работы в широковещательном режиме зарезервированы только три команды: начало передачи (NSTART), передача данных (NDATA), конец передачи (NEND). Ответы от конечных получателей в данном режиме не предусматриваются. На рис. 2 представлено схематическое изображение сессии передачи данных.

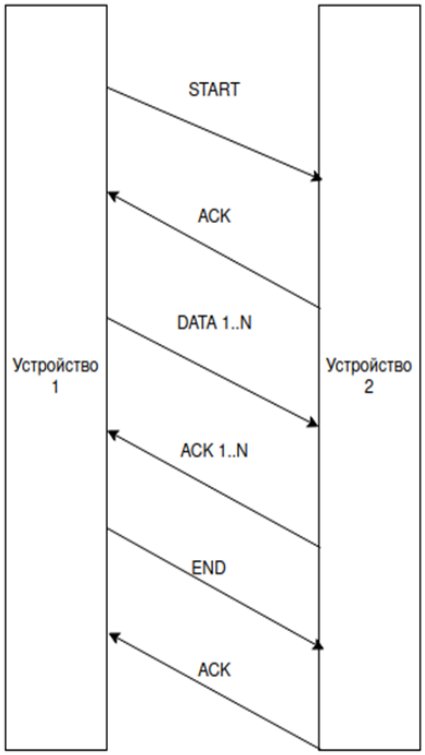

Рис. 2. Сессия передачи данных

Отправитель инициирует начало передачи, а получатель подтверждает готовность принять сообщение. В случае, если начало передачи невозможно по причине сбоя на стороне получателя, он может отклонить начало передачи (превышен лимит активных сессий и т. д.). Далее отправитель передает пакеты с данными, а получатель подтверждает успешный прием каждого пакета. Когда все данные переданы получателю, отправитель закрывает сессию. В случае сбоя получатель и отправитель могут прервать сессию передачи.

Подходы, заложенные в протокол, позволяют использовать все средства работы интерфейса *CAN*. Например, фильтрацию идентификаторов *CAN*–кадров, что позволяет игнорировать кадры, адресованные другим

устройствам на аппаратном уровне и уменьшить потребление вычислительных ресурсов. Это обеспечивается тем, что заголовок протокола располагается в идентификаторе *CAN*–кадра.

Также в библиотеке, реализующей данный протокол, обеспечивается возможность тонкой настройки для реализации максимальной производительности на конкретном устройстве. Пользователь библиотеки может выбрать тип выделения памяти: статический или динамический, идентификатор устройства, максимальную длину сообщения, количество буферов на отправку и прием сообщений, количество одновременно открытых сессий, время ожидания кадра в рамках сессии.

Тестирование библиотеки проводилось в виртуальной среде, структурная схема которой представлена на рис. 3.

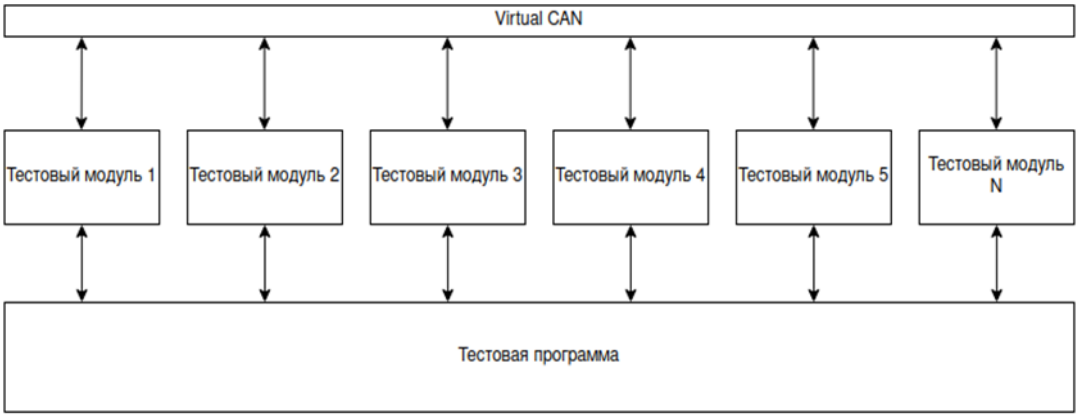

Рис. 3. Структурная схема тестовой виртуальной среды

Каждый программный модуль, подключенный к сети *Virtual CAN*, имитирует работу конечного устройства, тестовая программа следит за работой каждого модуля. Это позволяет значительно упростить отладку исходного кода и эмулировать ситуации, которые сложно воспроизвести в целевой системе, а также проверить работу библиотеки в предельных режимах работы.

ВЫВОД

Представленные в статье протокол и библиотека были протестированы на различных аппаратных платформах (STM32L4, Rapberry Pi, PC, ATSAM), успешно функционируют в BSUSat-2 и разработанной инженерной модели СМКА.

БИБЛИОГРАФИЧЕСКИЕ ССЫЛКИ

1. CAN Specification Version 2.0 // Robert Bosch GmbH, 1991. P.72.
2. EMR 3 CAN BUS specification Version 11-3//DEUTZ, 2005. P.53.