# Применение стека Simple MAC

## в беспроводных приложениях на базе микроконтроллеров семейства STM32W

**Денис Ягов**
yagov@promelec.ru

### Беспроводная система с ARM-контроллером на одном кристалле

В декабре 2010 г. компания STMicroelectronics официально начала массовый выпуск контроллера на базе ядра ARM Cortex M3 с интегрированным радиочастотным интерфейсом, построенным по стандарту IEEE 802.15.4/2,4 ГГц. На данный момент доступны два вида микроконтроллеров: STM32W108C8U6 и STM32W108CBU6. Оба они выполнены в одинаковом корпусе VFQFPN48 и полностью совместимы по выводам. Отличие заключается в объеме флэш-памяти: в первой модели 64 кбайт, во второй — 128 кбайт. В стадии испытаний находится модель STM32W108HBU6 — микроконтроллер в корпусе VFQFPN40 с 128 кбайт флэш-памяти и сниженным (по сравнению с существующими моделями) набором стандартной периферии.

Описание свойств ядра ARM Cortex M3 достойно отдельной книги. Мы отметим лишь то, что оно является естественным продолжением линейки ARM. Соответственно, и все свойства ядра ARM им унаследованы. Кроме того, имеется множество улучшений, например интегрированный в ядро контроллер векторных прерываний. Благодаря этому реакция на события и прерывания значительно выше, чем в ядрах ARM, а энергопотребление — ниже за счет исключения некоторых ненужных операций. Ядро ARM Cortex M3 способно выполнять 16- и 32-битные команды в едином потоке. Такой возможности не было прежде, поэтому разработчики были вынуждены заранее выбирать между низкой производительностью и высокой плотностью кода или высокой производительностью, но низкой плотностью кода. Есть множество и других преимуществ, но главное заключается в том, что ARM Cortex — это ядро будущего, которому предстоит потеснить другие ядра, в том числе и 8-битные. Уже сейчас разработчиков ждет великое множество сред разработки платных и бесплатных внутрисхемных отладчиков, библиотек, примеров приложений. Традиционно контроллеры с беспроводным интерфейсом выполнялись 8- и 16-разрядными на базе ядер 8051. STM32W — это первый массово выпускаемый контроллер с ядром ARM Cortex M3.

Отметим, что TI, Atmel и другие компании также объявили о планах выпуска контроллера с аналогичным радиоинтерфейсом на базе ядра ARM Cortex. Желание множества производителей выпускать микроконтроллеры с беспроводным интерфейсом в паре с мощным ядром можно назвать тенденцией развития беспроводных технологий.

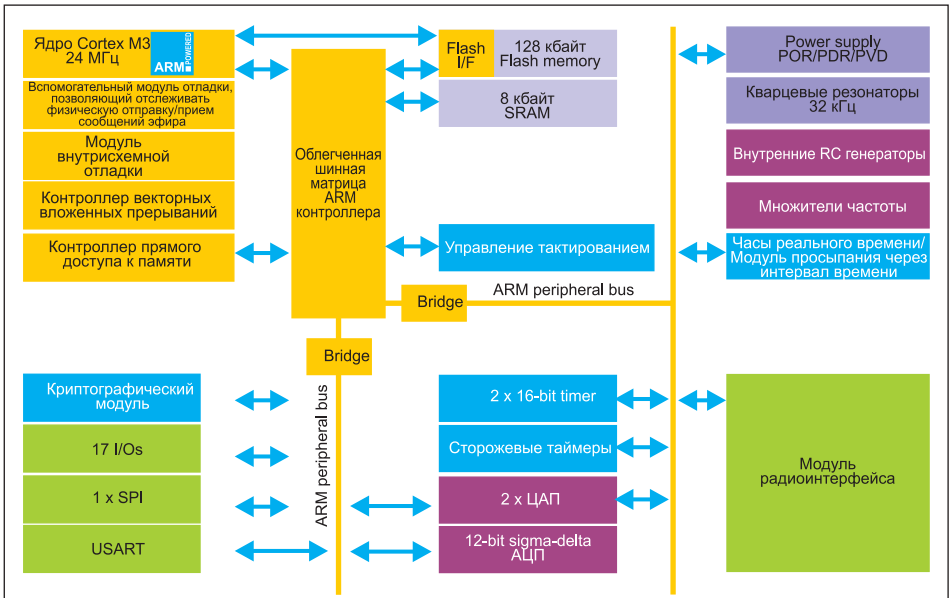


**Рис. 1.** Архитектура микроконтроллера STM32W

### Свойства микроконтроллеров STM32W

Любое беспроводное решение должно обладать низким энергопотреблением. В противном случае нужно будет привязывать его к проводам питания, что, безусловно, противоречит самой идее построения приложения без проводов. Таким образом, контроллеру требуется необходимый минимум стандартной и вспомогательной периферии, обслуживающей обмен по радиоканалу, которая, с одной стороны, разгрузит ядро и внутренние обмены данными, с другой — снизит энергопотребление в сравнении с решением аналогичной задачи софтом. К вспомогательной периферии радиочастотного интерфейса можно отнести аппаратные криптографы, вычислители контрольных сумм, внутренние модули отладки и т. п. Архитектура семейства STM32W показана на рис. 1.

В таблице 1 приведено сравнение беспроводных решений различных производителей.

**Таблица 1.** Сравнение беспроводных решений от Atmel, TI и STM

| Решение (Производитель) | Передача | | Прием | | Потери СВЧ-тракта, дБ |
|---|---|---|---|---|---|
| | Потребление, мА (мощность 3 дБ) | Максимальная мощность, дБ | Потребление, мА | Чувствительность, дБ | |
| CC2530 (TI) | 32 | 4,5 | 22 | –97 | – |
| STM32W108 (ST) | 26 | 8 | 22 | –100 | – |
| Atmega 128RFA1 (Atmel) | 14,5 | 3,5 | 12,5 | –96 | 10 |

**Таблица 2.** Сравнение параметров Atmega128RFA1, STM32W и CC2530

| Решение | Типовой ток потребления, мкА | Условия измерения |
|---|---|---|
| CC2530 | 1 | Напряжение питания 3 В; температура +25 °C; таймер просыпания работает; внешний часовой кварцевый резонатор; сохранение контекста памяти и регистров. |
| STM32W108 | 0,7 | Напряжение питания 3,6 В; температура +25 °C; таймер просыпания работает; низкоскоростной RC-генератор — источник тактирования таймера просыпания; содержимое ОЗУ и регистров сохраняется. |
| | 1,3 | Напряжение питания 3,6 В; температура +25 °C; таймер просыпания работает; часовой кварцевый резонатор — источник тактирования таймера просыпания; содержимое ОЗУ и регистров сохраняется. |
| Atmega128RFA1 | 0,25 | Напряжение питания 3 В; температура +25 °C, WDT отключен; содержимое ОЗУ и регистров сохраняется. |

Больше всего энергии потребляет радиоинтерфейс. Значительную часть времени контроллер должен находиться в спящем режиме. Крайне желательно, чтобы приемник был выключен на время «сна». Причина очевидна: типовое потребление приемника составляет десятки миллиампер, что, безусловно, противоречит идее батарейного питания прибора. Решение данного вопроса лежит в разделении времени работы узла сети на «сон» и работу в эфире (вне зависимости от того, что будет производиться: передача или прием сигнала). Естественно, все узлы сети должны одновременно начинать работу в эфире. Какой смысл, если один узел сети будет транслировать информацию, когда остальные его не слышат. Сеть должна быть синхронизирована по времени работы — первое, что необходимо для низкого энергопотребления сети. Синхронизация сети, безусловно, усложняет процесс обмена по беспроводному каналу, однако позволяет вводить в энергосберегающий режим не только ядро контроллера, но и радиочастотный модуль. Таймер, по которому для проведения радиообмена должны просыпаться ядро и необходимая периферия, производители, как правило, делают специальный. Его особенность — тесная интеграция с модулем беспроводной связи. Благодаря этому свойству беспроводная сеть может сама синхронизироваться, без постоянного вмешательства ядра в этот процесс.

Во вторую очередь необходимо обеспечить низкое потребление энергии в спящем режиме. В таблице 2 сравниваются параметры Atmega128RFA1, STM32W и CC2530.

Сеть можно настроить таким образом, что обмен будет происходить один раз в час. Такая сеть могла бы использоваться в системах сбора данных счетчиков тепла, воды, газа и т. п. Естественно, обмен информацией между узлами сети составит доли секунды, а остальное время узлы сети будут находиться в энергосберегающем режиме. Очень важно, чтобы все это время беспроводное устройство не расходовало энергию. Эффективность этого метода будет определяться током потребления в режиме сна.

Третье, что необходимо для низкого энергопотребления беспроводного устройства, — настраиваемая периферия, которая может работать с отключенным ядром. Заметно снизить энергопотребление позволяет контроллер прямого доступа к памяти. Если потребуется производить хоть сколько-нибудь интеллектуальные вычисления, не пробуждая ядро, то именно он позволит это сделать. Например, счетчик тепла при получении импульса датчика оборотов должен получить показания датчиков температуры теплоносителя на входе и выходе радиатора. Можно для этой цели использовать ядро: каждый раз при получении импульса оно будет вычислять разность температур и расход теплоносителя, рассчитывать на основании этих данных тепловые потери и суммировать с общим расходом тепла. А можно пойти другим путем: контроллер прямого доступа к памяти будет производить копирование результатов преобразования по приходу импульса датчика оборота. Ядро при этом находится в спящем режиме. Через определенное время ядро необходимо пробудить и обработать массив измерений. Второй метод может оказаться более эффективным с точки зрения потребления энергии.

STM32W имеет в своем составе контроллер прямого доступа к памяти DMA. Отметим некоторые свойства DMA в семействе STM32W. Во-первых, DMA может работать с радиочастотным модулем: отправлять и принимать сообщения. При этом копирование принятого сообщения осуществляется после фильтрации и аппаратной проверки контрольной суммы, если она совпала. Контроллер DMA может осуществлять копирование по запросам с последовательных интерфейсов SPI, I²C и UART аналого-цифрового преобразователя.

Стоит отметить, что имеется ряд других периферийных устройств, которые также позволяют снизить энергопотребление контроллера, например модуль аппаратной криптографии AES128, модуль вычисления контрольной суммы, модуль генерации случайных чисел. Все они имеют несомненное превосходство в энергопотреблении перед программным решением аналогичных задач.

## Модуль радиопередачи в STM32W

Рассмотрим более подробно модуль (рис. 2), который непосредственно отвечает за передачу и прием сообщений по радиоканалу. В состав данного модуля входят:

- Высокочастотный осциллятор (HF OSC) с внешним кварцевым резонатором.
- Синтезатор частоты, состоящий из детектора фаз (PFD), который через фильтр выдает

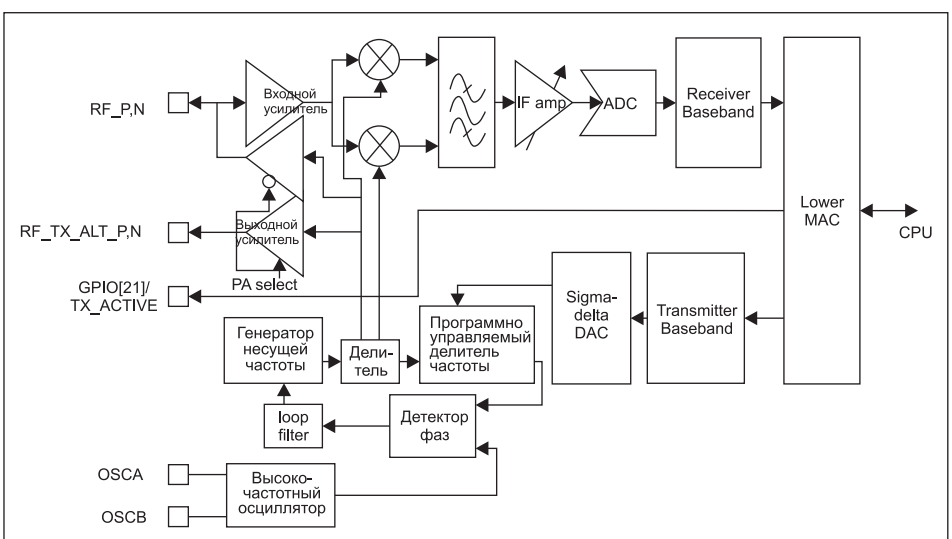


**Рис. 2.** Схема модуля радиочастотного интерфейса

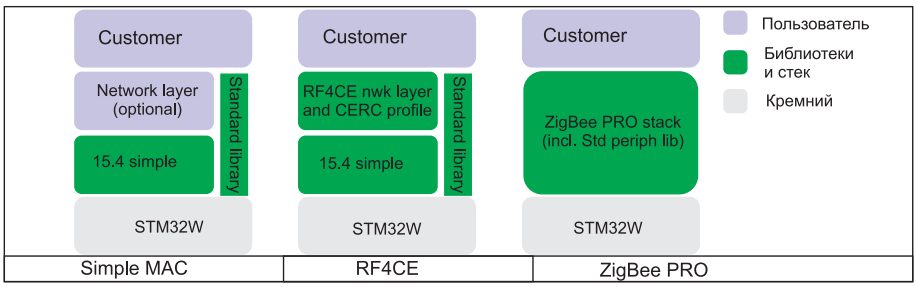


**Рис. 3.** Программные стеки для микроконтроллеров STM32W

**Таблица 3.** Преимущества радиочастотной технологии

| Передача данных ИК-лучами | Передача данных посредством RF4CE |
|---|---|
| Требуется открытое пространство между приемником и передатчиком | Не требуется открытое пространство между приемником и передатчиком |
| Связь односторонняя без подтверждений | Двусторонняя связь |
| Плазменные панели и подсветка TFT подавляют ИК-сигнал | Быстрая передача сигнала с дополнительными функциями (подтверждение приема) |
| Множественная отправка команд | Однократная отправка, требуется на 25% меньше энергии |
| Каждый продукт имеет свои собственные команды | Все стандартизовано |

управляющее напряжение на генератор несущей частоты (VCO). Выходная частота 4,8 ГГц делится (divider) на 2 и попадает на выход синтезатора частоты (prescaler) и на программно управляемый делитель частоты, откуда снова на детектор сдвига фаз. Таким образом, синтезатор частоты генерирует высокоточный радиочастотный сигнал 2,4 ГГц.

- Набор усилителей радиосигналов: входного (LNA); выходного, совмещенного с приемной антенной (PA); выходного для подключения к раздельной передающей антенне либо к внешнему радиочастотному усилителю. Возможность перевода внешнего радиочастотного усилителя в режим низкого энергопотребления также предусмотрена, для чего MAC-контроллер применяет линию порта ввода/вывода.
- Модуль квадратурной обработки входного сигнала и фильтрации. На выходе данного модуля входной сигнал с частотой 2,4 ГГц преобразуется в сигнал 4 МГц.
- Усилитель с подстраиваемым коэффициентом усиления, управляемым MAC-уровнем, и высокоскоростной АЦП (12 Мвыб/с).
- MAC-контроллер, осуществляющий первоначальную обработку принятого цифрового сигнала.

## Виды стеков для микроконтроллеров STM32W и их сравнение

На текущий момент существует как минимум три стека беспроводной передачи данных для контроллеров семейства STM32W (рис. 3). Это Simple MAC, RF4CE и ZigBee PRO. На основе стека Simple MAC можно реализовать любой низкоскоростной обмен данными в сети произвольной топологии. При этом обслуживание сетевого уровня ложится непосредственно на разработчика приложения. Созданное приложение будет соответствовать стандарту беспроводной передачи данных IEEE 802.15.4/2,4 ГГц. Именно гарантия соответствия этому стандарту является самой веской причиной для использования данного стека. Стек RF4CE (Radio Frequency For Consumer Electronics — радиочастота для потребительской электроники) является надстройкой стека Simple MAC.

Стек RF4CE, главным образом, предназначен для замены ИК-пультов управления на радиочастотные. Есть несколько неоспоримых преимуществ радиочастотной технологии построения пульта управления перед классической, на базе инфракрасных источников/приемников. Основные преимущества радиочастотной технологии показаны в таблице 3.

В конечном счете, передача команд управления по радиоканалу более технологична, удобна и экономна с точки зрения энергопотребления. Если говорить о цене решения, то даже в нынешних условиях передача данных по радиоканалу может быть соизмерима по цене с классической инфракрасной технологией. Цену радиочастотной технологии снижают отсутствие требований конструкции корпуса источника и приемника, а также упрощение технологического процесса производства. Если в будущем начнется массовый выпуск радиочастотных пультов управления потребительской электроники, то цена технологии еще больше упадет, что приведет к отказу применения технологии ИК-передачи данных.

Стек ZigBee PRO организует сразу всю сеть ZigBee PRO, в которой каждое устройство может выполнять одну из ролей: конечный источник/приемник данных, маршрутизатор или координатор всей сети. Такая сеть отличается высокой надежностью и защищенностью. Это достигается за счет дополнительных методов маршрутизации, которые выявляют ненадежные асимметричные каналы связи, и возможностью смены частот для отдельных узлов. Такая сеть может передавать длинные сообщения, разбивая их на фрагменты, и ее разумно применять для автоматизации зданий, интеллектуального управления освещением, мониторинга состояния пациента, в системах энергоучета и т. п.

## Стек Simple MAC

Как говорилось ранее, данный стек является самым простым и гарантирует соответствие приложений пользователя стандарту IEEE 802.15.4/2,4 ГГц. Все команды стека Simple MAC можно разбить на несколько групп:

- команды инициализации и пробуждения;
- команды выбора канала, мощности передачи;
- команды трансляции;
- команды приема;
- команды MAC-таймера;
- остальные.

Стек Simple MAC обменивается пакетами в соответствии со стандартом IEEE 802.15.4 (рис. 4).

Пакет обмена выглядит так:

- 4 байт синхронизации;
- команда начала данных;
- длина передаваемых данных (максимум 128 байт);
- данные (пакет Simple MAC).

Итак, для обмена данными через эфир по стандарту IEEE 802.15.4 достаточно передавать пакет с данными длиной не более 127 байт указанного выше вида. При этом, как видно из структуры пакета, стандарт никак не регламентирует адресацию пакета, достоверность доставляемых данных (контрольную сумму), разбиение длинных пакетов на мелкие и прочие свойства, характерные для сетевой передачи. По сути, стандартом IEEE 802.15.4 объявлен физический и частично канальный уровень обмена модели OSI.

| 4 bytes | 1 byte | 1 byte | | Variable |
|---|---|---|---|---|
| Preamble | SFD | Frame length (7 bits) | Reserved (1 bit) | PHR payload filed (PSDU) |
| SHR | | PHR | | PHY payload |

**Рис. 4.** Вид пакета данных согласно стандарту IEEE 802.15.4

| 2 bytes | 1 bytes | 0/2 bytes | 0/2/8 bytes | 0/2 bytes | 0/2/8 bytes | Variable | 2 bytes |
|---|---|---|---|---|---|---|---|
| Поле контроля пакета | Номер пакета | Глобальный адрес назначения | Локальный адрес назначения | Глобальный адрес источника | Локальный адрес источника | Данные пакета | Контрольная сумма пакета |
| | | Addressing fields | | | | | |
| MHR | | | | | | MAC payload | MFR |

**Рис. 5.** Структура пакета данных

Стандарт Simple MAC является надстройкой стандарта IEEE 802.15.4 и реализует дополнительные функции беспроводного сетевого обмена: во-первых, адресацию внутри локальной сети; во-вторых, адресацию локальных сетей; в-третьих, достоверность доставки пакета. Simple MAC является информационной частью указанного выше пакета и, в свою очередь, имеет свою строго определенную структуру (рис. 5).

Поле FCF задает свойства пакета с последующей информацией:

- наличие/отсутствие адреса локальной сети источника;
- длина адреса источника;
- наличие/отсутствие адреса локальной сети приемника;
- длина адреса приемника;
- требуется или нет подтверждающее сообщение приемника;
- требуется или нет шифрование данных;
- тип пакета.

Остальные поля — это номер пакета, адреса источника и приемника пакета (включая адрес сети), сам пакет и контрольная сумма. На данный момент стек реализует четыре вида пакетов обмена информацией:

- сигнальный (beacon);
- обмена данными;
- подтверждение приема;
- команда MAC.

Как говорилось ранее, тип пакета находится в поле FCF. В простейшем случае построения обмена на базе надстройки Simple MAC в пакет стандарта IEEE 802.15.4 обязательно добавятся 5 байт: 2 байта FCF, 1 байт — номер пакета, контрольная сумма — 2 байта. Можно построить обмен только пакетами такого вида. При этом все посылки будут широковещательными, приемник данных будет лишен возможности обратной связи с источником, номер пакета в посылке, по большому счету, будет лишней нагрузкой (как вариант, можно этот байт использовать приемником для статистики пропущенных пакетов). Однако данные, принятые приемником, можно будет считать достоверными, так как их набору будет соответствовать определенная контрольная сумма. Некоторым приложениям достаточно даже такого обмена.

На мой взгляд, гораздо более интересным выглядит применение прочих возможностей стека. Начнем с адресации. Каждому устройству можно назначить адрес внутри локальной сети и адрес самой локальной сети. Каким образом будут назначены адреса — дело разработчика: они могут быть прошиты «железно», а могут получаться от устройства с неким «волшебным адресом». Кроме самой трансляции стек Simple MAC реализует фильтрацию. Приложение не увидит тех сообщений, которые не пройдут фильтр. Фильтрация возможна по адресам сети и адресам устройств внутри сети. Получив сообщение, которое прошло фильтр адресов и проверку контрольной суммы, приложение может выслать подтверждение приема, если в принятом пакете в поле FCF указана необходимость его отправки. Получив пакет специального вида, источник данных удостоверяется в успехе процедуры трансляции.

Стек Simple MAC позволяет сделать отправку подтверждения приема автоматической. Как будет себя вести источник данных, если подтверждение не пришло, решает разработчик. Вместе с тем стек позволяет несколько автоматизировать процесс: можно ввести количество повторных отправок и время, по истечении которого производить первую повторную отправку. Вместе с принимаемым сообщением приложение получает уровень сигнала приема. Соответственно, пользователь должен реагировать на эту информацию и в случае необходимости изменять чувствительность приемника.

Кроме пакетов обмена данными и подтверждений, имеются пакеты обмена информацией между MAC-уровнями, например команды синхронизации следующего сеанса связи. Пользователь не обязательно их видит, однако он может настроить средствами стека Simple MAC связь таким образом, что узлы будут отключать приемники на определенное время между сеансами связи. Как показано ранее, это значительно снизит энергопотребление.

Итак, что такое Simple MAC? В целом, это хорошая основа для создания собственного протокола беспроводного обмена. Стек выполняет большинство рутинных операций, востребованных во время связи. Вид сети, иерархию, возможность ретрансляции на следующие узлы и т. п. определяет разработчик приложения. Ограничения применения, по большому счету, заложены стандартом IEEE 802.15.4 и Simple MAC — это размер и вид пакета.

По мнению автора, Simple MAC является хорошей альтернативой «открытой платформе IEEE 802.15.4». Главные его достоинства:

- минимальный, но достаточный набор команд;
- полная документация;
- примеры применения.

Начиная проект с «открытой платформой IEEE 802.15.4» на контроллере семейства STM32W, всего этого разработчик пока лишен. Минусом применения стека можно считать потенциальную возможность уменьшить объем кода за счет собственной оптимизации и упрощения протокола обмена.

## Простейшее приложение на базе стека Simple MAC

На каждый стек компания STMicroelectronics выпустила по нескольку примеров. В целях демонстрации возможностей микроконтроллеров семейства STM32W они допускают возможность подключения разной периферии. Одновременно в одном приложении может быть задействована почти вся периферия. Это, безусловно, демонстрирует возможности контроллера, но затрудняет понимание работы самой интересной части — радиочастотного модуля.

Мы строили свое приложение на базе стандартного примера применения стека Simple MAC, исключив при этом практически все лишнее и коды. Задача была простая: передать беспроводным способом с одной платы на другую пакет, который будет символизировать, что на передающем модуле нажата кнопка. Таким образом, программа практически не содержит того, что могло бы отвлечь внимание от работы с радиочастотным интерфейсом. Объем кода программы в результате таких действий упал с 30 до 10 кбайт. Наиболее важные моменты этого проекта приведены в листинге.

Во-первых, система обязана тактироваться от внешнего кварца и должны быть разрешены прерывания (для работы беспроводного интерфейса):

```
//установка клокинга
halInternalSetRegTrim(FALSE);
halInternalSwitchToXtal();
halInternalCalibrateFastRc();
INTERRUPTS_ON();
```

Во-вторых, необходимо провести инициализацию радиомодуля и установить адрес устройства и адрес сети (в нашем случае адрес 0x1604):

```
//инициализация радиомодуля
assert(ST_RadioInit(ST_RADIO_POWER_MODE_RX_ON)==
==ST_SUCCESS);
//установка адреса устройства в сети и адреса сети
ST_RadioSetNodeId(0x1604);
ST_RadioSetPanId(0x1604);
```

Затем выделить память под буферы приема и отправки сообщений:

```
#pragma align txPacket
u8 txPacket[128] = {
0x0a, // length
0x61, // fcf - intra pan, ack request, data
0x08, // fcf - src, dst mode
0x00, // seq
0x04, // нижний байт адреса сети
0x16, // верхний байт адреса сети
0x04, // нижний байт адреса абонента сети
0x16, // верхний байт адреса абонента сети
0x00 // data
};
// буфер для получаемого пакета
u8 rxPacket[128];
// флаг «пакет принят»
boolean packetReceived = FALSE;
```

Отправляемый пакет должен содержать длину пакета; меняющийся с каждой отправкой номер пакета (см. структуру пакета); адреса источника и приемника (в нашем примере они прописаны выше).

Отправка пакета:

```
// длина пакета состоит из 7 байт преамбулы + 2 CRC +
передаваемые данные
txPacket[0] = 16 + 2 + 7;
// номер пакета, должен отличаться с каждой посылкой
txPacket[3]++;
//txComplete = FALSE;
ST_RadioTransmit(txPacket);
```

При успешном приеме пакета вызывается процедура ST_RadioReceiveIsrCallback, тело которой может записать разработчик ПО. В нашем случае производится копирование принятых данных в ОЗУ. Количество копируемых данных указано в пакете (packet[0]). Переменная packetReseved — внутренняя переменная приложения.

Прием пакета:

```
void ST_RadioReceiveIsrCallback(u8 *packet,
boolean ackFramePendingSet,
u32 time,
u16 errors,
s8 rssi)
{
// данная процедура вызывается в контексте прерывания
u8 i;
// копирование данных пакета в заданную область памяти
if(packetReceived == FALSE) {
for(i=0; i<=packet[0]; i++) {
rxPacket[i] = packet[i];
}
// флаг «Пакет принят»
packetReceived = TRUE;
}
}
```

Есть аналогичные функции, которые будут вызываться в процессе работы радиомодуля и которые также можно наполнить своим кодом. Таким образом, можно создать приложение, которое будет весьма гибко реагировать на различные ситуации, связанные с отправкой и приемом сообщений через эфир.

## Заключение

Мы рассмотрели элементарный пример применения стека Simple MAC. Как видно из основных частей листинга, работа с радиочастотным интерфейсом не сложнее, чем, например, с UART. Организовать соединение «точка-точка» не составляет большого труда. Применение этого стека не требует слишком высокой квалификации разработчика. Интеграция радиочастотного модуля в приложение также не является сложной задачей. Сравнить инициализацию беспроводного обмена с организацией обмена по UART можно в разделе «Примеры применения» [1].

Архиважный вопрос для применения данной технологии — ее стоимость. Как было отмечено выше, цена этой технологии будет падать и через какое-то время станет соизмеримой с существующей ИК-технологией передачи данных. ■

## Литература


1. Материал тренинга для STM32W: http://www.promelec.ru/stm/stm32/
2. Обзор семейства контроллеров STM32W: http://www.st.com/internet/mcu/subclass/1377.jsp
3. Описания контроллеров STM32W: http://www.st.com/stonline/stappl/resourceSelector/app?page=resourceSelector&doctype=DATASHEET&SubClassID=1377
4. Стеки и примеры применения для контроллеров STM32W: http://www.st.com/stonline/stappl/resourceSelector/app?page=resourceSelector&doctype=FIRMWARE&SubClassID=1377
5. Описание работы Simple MAC: http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/USER_MANUAL/CD00262339.pdf
6. Ссылка для скачивания бесплатной среды для программирования контроллеров STM32W: http://supp.iar.com/Download/SW/?item=EWARM-KS32