Применение стэка Simple MAC в беспроводных приложениях на базе микроконтроллеров семейства STM32W.

1. Беспроводная система с ARM контроллером на одном кристалле.

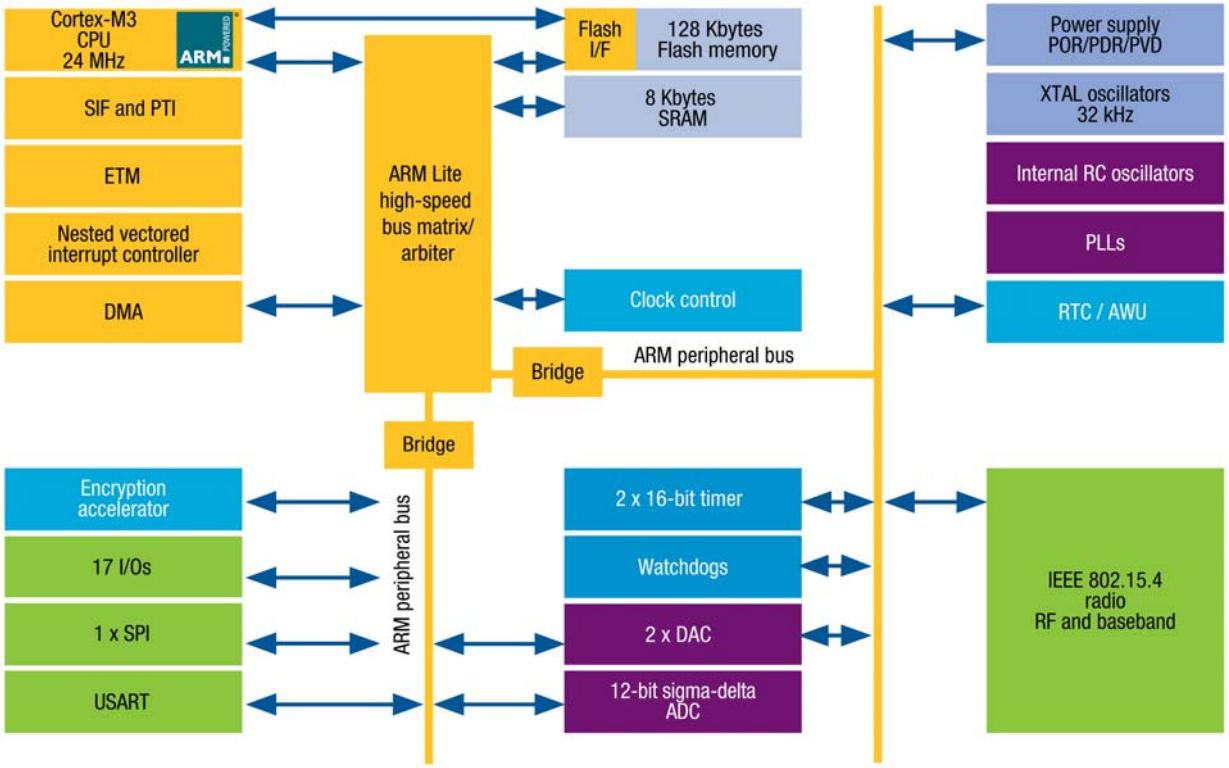
В декабре 2010 года компания STMicroelectronics официально начала массовый выпуск контроллера на базе ядра ARM-Cortex M3 с интегрированным радиочастотным интерфейсом построенному по стандарту IEEE 802.15.4 / 2.4ГГц. На данный момент доступны 2 вида микроконтроллеров – это STM32W108C8U6 и STM32W108CBU6. Оба микроконтроллера выполнены в одинаковом корпусе VFQFPN48, полностью совместимы по выводам. Отличие заключается в объёме памяти флэш: в первой модели – 64кб, во второй – 128кб. В стадии испытаний находится модель STM32W108HBU6 – микроконтроллер в корпусе VFQFPN40, с 128кб флэш памяти и сниженным набором стандартной периферии в сравнении с существующими моделями.

Главное отличие данного контроллера с беспроводным интерфейсом заключается в том, что построен он на базе ядра ARM Cortex M3. Описание свойств ядра ARM Cortex M3 достойно отдельной книги. Мы отметим лишь то, что это ядро является естественным продолжением линейки ядер ARM. Соответственно, все свойства ядра ARM унаследованы ядром ARM Cortex. 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, энергопотребление – ниже за счёт исключения некоторых ненужных операций. Ядро ARM Cortex M3 способно выполнять 16-ти и 32- битные команды в едином потоке. Такой возможности не было на ядрах ARM, поэтому разработчики заранее выбирали между низкой производительностью, но высокой плотностью кода или высокой производительностью, но низкой плотностью кода. Есть множество других преимуществ, но главное из них – ARM Cortex – это ядро будущего, которое сильно потеснит другие ядра, в том числе и 8-ми битные. Уже сейчас разработчиков на данном ядре ждёт великое множество сред разработки платных и бесплатных, внутрисхемных отладчиков, библиотек, примеров приложений. Традиционно контроллеры с беспроводным интерфейсом – это 8-ми и 16-ти разрядные контроллеры на базе ядер таких как 8051, но никак не ARM. STM32W – это первый массово выпускаемый контроллер с ядром ARM Cortex M3.

Отметим, что другие компании такие как TI, Atmel и др. так же объявили о планах выпуска контроллера с аналогичным радио интерфейсом на базе ядра ARM-Cortex. Желание множества производителей выпускать микроконтроллеры с беспроводным интерфейсом в паре с мощным ядром – тенденция развития беспроводных технологий.

2. Свойство микроконтроллеров STM32W

Любое беспроводное решение должно обладать свойством низкого энергопотребления. 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 периферии 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.

Периферией, которая больше всего потребляет энергии безусловно является радио интерфейс. Ниже приведена таблица сравнений беспроводных решений от TI и ST.

Решение	Передача		Приём		Потери СВЧ тракта (дБ)
	Потребление (мА) (мощность 3дБ)	Максимальная мощность (дБ)	Потребление (мА)	Чувствительность (дБ)	
CC2530	32	4.5	22	-97	-
STM32W108	26	8	22	-100	-
Atmega128RFA1	14.5	3.5	12.5	-96	10

Большую часть времени контроллер должен находиться в спящем режиме. Крайне желательно, чтобы приёмник был выключен на время сна. Причина – очевидна, типовое потребление приёмника составляет десятки миллиампер, что, безусловно, противоречит батарейному питанию 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 приём сигнала). Естественно, все узлы сети должны одновременно начинать работу в эфире. Какой смысл, если один узел сети будет транслировать информацию, когда остальные его не услышат. Сеть должна быть синхронизирована по времени работы – первый краеугольный камень 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 должно просыпаться ядро и необходимая периферия для проведения радиообмена, производители, как правило, делают специальный. Спецификой данного таймера является его тесная интеграция с модулем беспроводной связи. Благодаря этому свойству беспроводная сеть может сама синхронизироваться без постоянного вмешательства ядра в этот процесс.

Второй краеугольный камень беспроводных устройств – низкое потребление энергии в спящем режиме. Далее приведена таблица сравнения параметров STM32W и CC2530.

Решение	Типовой ток потребления	Условия измерения
CC2530	1 мкА	Напряжение питания 3.0В, Температура +25°С, Таймер просыпания работает, внешний часовой кварцевый резонатор, сохранение контекста памяти и регистров.
STM32W108	0.7 мкА	Напряжение питания 3.6В, Температура +25°С, Таймер просыпания работает, низкоскоростной RC-генератор – источник тактирования Таймера просыпания. Содержимое ОЗУ и регистров сохраняется
STM32W108	1.3мкА	Напряжение питания 3.6В, Температура +25°С, Таймер просыпания работает, часовой кварцевый резонатор – источник тактирования Таймера просыпания. Содержимое ОЗУ и регистров сохраняется
Atmega128RFA1	0.25мкА	Напряжение питания 3.0В, температура +25°С, WDT – отключен, содержимое ОЗУ и регистров - сохраняется

Сеть можно настроить таким образом, что обмен будет происходить 1 раз в час. Такая сеть могла бы использоваться в системах сбора данных счётчиков 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 всё это время беспроводное устройство не расходовало энергию. Эффективность этого метода будет определяться током потребления в режиме сна.

Третий краеугольный камень низкого потребления беспроводного устройства – настраиваемая периферия, которая может работать с отключенным ядром. Периферия, которая сильно снизит энергопотребление – контроллер прямого доступа к памяти. Если потребуется производить хоть сколько-нибудь интеллектуальные вычисления, при этом не пробуждать ядро, то контроллер прямого доступа к памяти может быть востребован. Например, счётчик тепла при получении импульса датчика оборотов должен получить показания датчиков температуры теплоносителя на входе и выходе радиатора. Можно для этой цели применять ядро: каждый раз при получении импульса ядро вычисляет разность температур, расход теплоносителя и вычисляет на основании этого тепловые потери и сложит с общим расходом тепла. Можно пойти другим путём: контроллер прямого доступа к памяти будет производить копирование результатов преобразования по приходу импульса датчика оборота. Ядро при этом находится в режиме сна. Через определённое время ядро необходимо пробудить и обработать массив измерений. Второй метод может оказаться более эффективным с точки зрения потребления энергии. STM32W имеет в своём 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 сообщение 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 периферий, которые так же снизят энергопотребление контроллера. Это модуль аппаратной криптографии AES128, модуль вычисления контрольной суммы, модуль генерации случайных чисел. Все эти периферии имеют несомненное превосходство в энергопотреблении перед программным решением аналогичных задач.

3. Модуль радиопередачи в STM32W

Рассмотрим более подробно модуль, который непосредственно отвечает за передачу и приём сообщений по радиоканалу. Данный модуль состоит из:

- высокочастотного осциллятора (HF OSC) а внешним кварцевым резонатором
- синтезатора частоты состоящего из детектора фаз (PFD), который через фильтр выдаёт управляющее напряжение на генератор несущей частоты (VCO). Выходная частота 4.8ГГц делится

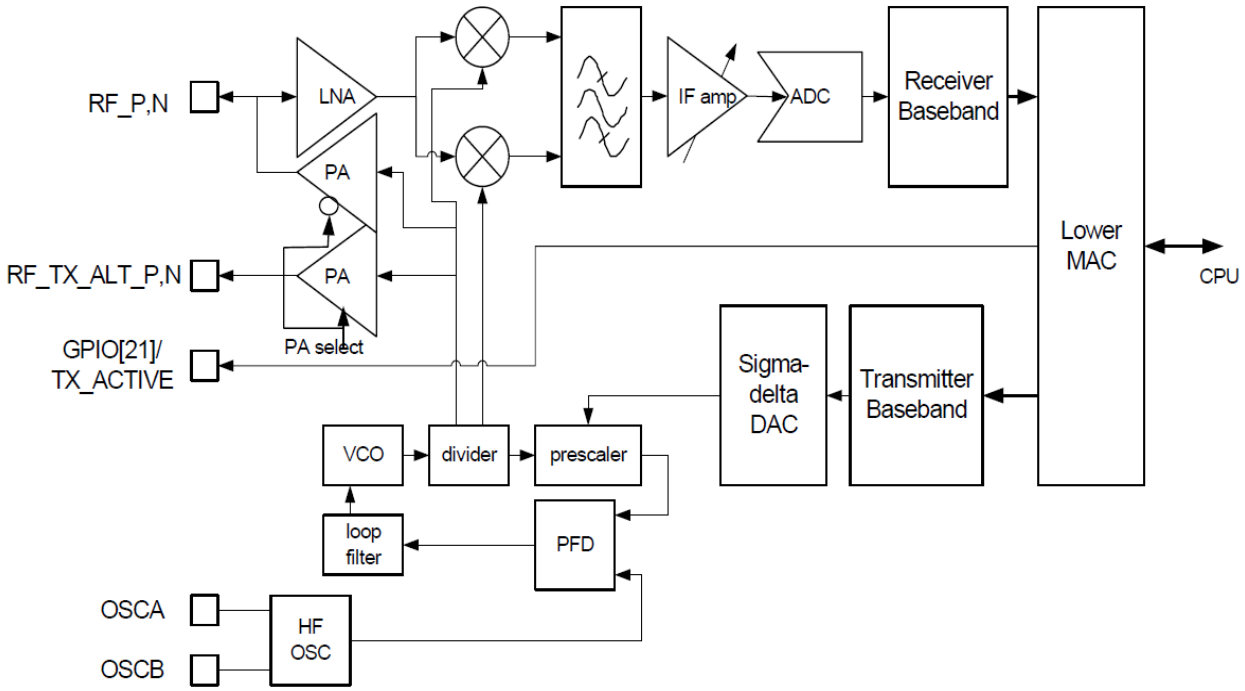
(divider) на 2 и попадает на выход синтезатора частоты (prescaler) и на программно управляемый делитель частоты, откуда снова на детектор сдвига фаз. Таким образом, синтезатор частоты генерирует высокоточный радиочастотный сигнал 2.4ГГц

- набора усилителей радиосигналов: входного (LNA), выходного, совмещённого с приёмной антенной (PA), выходного, для подключения к раздельной передающей антенне, либо к внешнему радиочастотному усилителю. Возможность перевода внешнего радиочастотного усилителя в режим низкого энергопотребления – так же предусмотрена, для чего MAC контроллер применяет линию порта ввода-вывода.

- модуля квадротурной обработки входного сигнала и фильтрации. На выходе данного модуля входной сигнал с частотой 2.4ГГц преобразуется в сигнал 4МГц.

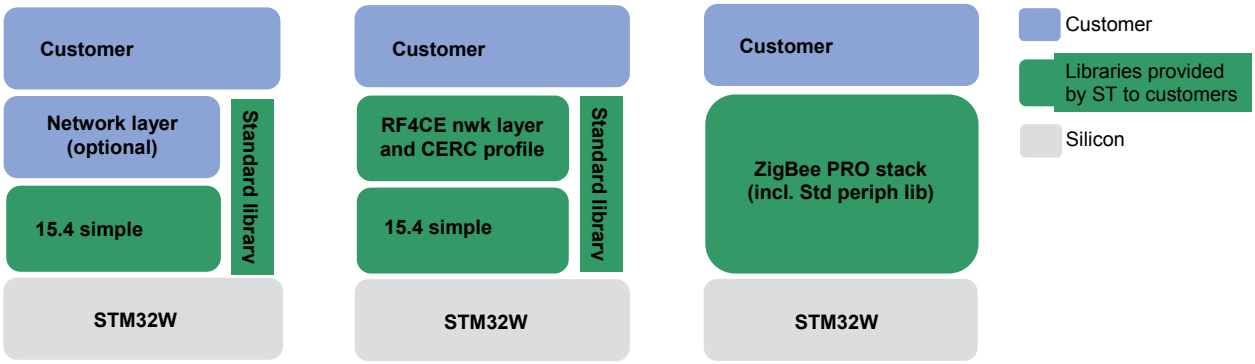
- Усилителя с подстраиваемым коэффициентом усиления, управляемым MAC уровнем и высокоскоростного АЦП (12Мвыб/сек).

- MAC контроллер осуществляет первоначальную обработку принятого цифрового сигнала.



4. Существующие виды стеков для микроконтроллеров STM32W и их сравнение.

На текущий момент существует как минимум 3 стека беспроводной передачи данных для контроллеров семейства STM32W. Это Simple MAC, RF4CE и ZigBee PRO. На основе стека Simple MAC можно создать любой низкоскоростной обмен данными, в сети произвольной топологии. При этом обслуживание сетевого уровня ложится непосредственно на разработчика приложения. При этом, созданное приложение будет соответствовать стандарту беспроводной передачи данных IEEE 802.15.4 / 2.4Ghz**.** Именно гарантия соответствия этому стандарту является самой веской причиной для использования данного стека. Стек RF4CE, который можно расшифровать, как Radio Frequency For Consumer Electronics (радио частота для потребительской электроники), является надстройкой стека Simple MAC



Simple MAC RF4CE ZigBee PRO

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/приёмников. Основные преимущества радиочастотной технологии показаны в следующей таблице.

Передача данных ИК лучами	**Передача данных посредством RF4CE**
Требуется открытое пространство между приёмником и передатчиком	Не требуется открытое пространство между приёмником и передатчиком
Связь односторонняя без подтверждений	Двусторонняя связь
Плазменные панели и подсветка TFT подавляют ИК сигнал	Быстрая передача сигнала с доп функциями (подтверждение приёма)
Энергопотребление: Множественная отправка команд	Однократная отправка 25% меньше энергии требуется
Каждый продукт имеет свои собственные команды	Всё стандартизовано

В конечном счёте, передача команд управления по радиоканалу более технологична, удобна и экономна, 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 приёмника, а так же упрощение технологического процесса производства. Если в будущем начнётся массовый выпуск пультов управления потребительской электроники, то цена технологии ещё упадёт, что приведёт к отказу применения технологии ИК передачи данных в пользу последней.

Стек ZigBee PRO. Данный стек организует сразу всю сеть ZigBee PRO, в которой каждое устройство может выполнять одну из ролей: конечный источник/приёмник данных, маршрутизатор или координатор всей сети. Такая сеть отличается высокой надёжностью и защищённостью. Это достигается за счёт дополнительных методов маршрутизации, которые выявляют ненадёжные, ассиметричные, каналы связи, возможностью смены частот для отдельных узлов. Такая сеть может передавать длинные сообщения, разбивая их на фрагменты. Такую сеть можно применить для автоматизации зданий, интеллектуального управления освещения, мониторинга состояния пациента, системах энергоучёта и других подобных системах.

5. Стек Simple MAC

Как говорилось ранее, данный стек является самым простым и гарантирует соответствие приложение пользователя стандарту IEEE 802.15.4 / 2.4Ghz. Все команды, стека Simple MAC можно разбить на несколько групп:

- команды инициализации и пробуждения
- команды выбора канала, мощности передачи

- команды трансляции
- команды приёма
- команды MAC таймера
- остальные команды

Стек Simple MAC обменивается пакетами в соответствии со стандартом IEEE 802.15.4. Пакет обмена выглядит так:

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

- 4 байта сихронизаци
- команда начала данных
- длина передаваемых данных (максимум 128байт)
- данные (пакет SimpleMAC)

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 Ози.

Стандарт SimpleMAC является надстройкой стандарта IEEE 802.15.4. Эта надстройка реализует дополнительные функции беспроводного сетевого обмена. Во-первых, реализует адресацию внутри локальной сети, во-вторых, адресацию локальных сетей, в-третьих – достоверность доставки пакета. Пакет SimpleMAC является информационной частью указанного выше пакета и, в свою очередь, имеет свою строго определённую структуру:

<table>
<tr><th>2 bytes</th><th>1 bytes</th><th>0/2 bytes</th><th>0/2/8 bytes</th><th>0/2 bytes</th><th>0/2/8 bytes</th><th>Variable</th><th>2 bytes</th></tr>
<tr><td rowspan="2">Frame control field (FCF)</td><td rowspan="2">Sequence number</td><td>Destination PAN identifier</td><td>Destination address</td><td>Source PAN identifier</td><td>Source address</td><td rowspan="2">Frame payload</td><td rowspan="2">FCS</td></tr>
<tr><td colspan="4">Addressing fields</td></tr>
<tr><td colspan="6">MHR</td><td>MAC payload</td><td>MFR</td></tr>
</table>

Поле FCF – задаёт свойства пакета с последующей информацией:
- наличие/отсутствие адреса локальной сети источника
- длину адреса источника
- наличие/отсутствие адреса локальной сети приёмника
- длину адреса приёмника
- требуется или нет подтверждающее сообщение приёмника
- требуется или нет шифрование данных
- тип пакета

Остальные поля – это номер пакета, адреса источника и приёмника пакета (включая адрес сети), сам пакет и контрольная сумма. На данный момент стек реализует 4 вида пакетов обмена информацией:
- Сигнальный (beacon)
- обмена данными
- подтверждение приёма
- 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 и контрольная сумма 2 байта. Можно построить обмен только пакетами такого вида. При этом, все посылки будут широковещательными, источник данных будет лишён возможности обратной связи с источником, номер пакета в посылке – по большому счёту будет лишней нагрузкой (как вариант, можно этот байт использовать приёмником для статистики пропущенных пакетов), однако данные принятые приёмником можно будет считать достоверными, т.к. набору принятых данных будет соответствовать определённая контрольная сумма. Некоторым приложениям достаточно даже такого обмена. На мой взгляд, гораздо более интересным выглядит применение остальных возможностей стека. Начнём с адресации. Каждому устройству можно назначить адрес внутри локальной сети и адрес самой локальной сети. Каким образом будут назначены адреса – дело разработчика сети: адреса 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 приёма, если в принятом пакете в поле FCF указана необходимость отправки пакета подтверждения приёма. Получая пакет специального вида, источник данных удостоверяется в успехе процедуры трансляции. Стек Simple Mac позволяет сделать отправку подтверждения приёма автоматической. Как будет себя вести источник данных, если подтверждение не пришло – решает разработчик. Вместе с тем, стек позволяет несколько автоматизировать процесс: можно ввести параметр количества повторных отправок и время, по истечению которого, производить первую повторную отправку.

Вместе с принимаемым сообщением, приложение получает уровень сигнала приёма. Соответственно, пользователь должен реагировать, и в случае необходимости изменять чувствительность приёмника.

Кроме пакетов обмена данными и подтверждений имеются пакеты обмена информацией между Mac уровнями. Например, команды синхронизации следующего сеанса связи. Пользователь может не видеть пакетов синхронизации, однако, может настроить средствами стека Simple Mac связь таким образом, что узлы будут отключать приёмники на определённое время, между сеансами связи. Как показано ранее, это значительно снизит энергопотребление.

Итак, что такое Simple MAC? По большому счёту – это хорошая основа для создания собственного протокола беспроводного обмена. Стек выполняет большинство рутинных операций востребованных во время связи. Определение вида сети, иерархии, возможности ретрансляции на следующие узлы и т.п. определяет разработчик приложения. Ограничения применения, по большому счёту, заложены стандартом IEEE 802.15.4 и Simple Mac – это размер и вид пакета.

По мнению автора, стек Simple Mac является хорошей альтернативой «открытой платформе IEEE 802.15.4». Главными достоинствами являются:

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

Всего этого разработчик пока лишён, начиная проект с «открытой платформой IEEE 802.15.4» на контроллере STM32W семейства. Минусом применения стека можно считать потенциальную возможность уменьшить объём кода за счёт собственной оптимизации и упрощения протокола обмена.

6. Простейшее приложение на базе стека Simple MAC

На каждый стек компания STMicroelectronics выпустила по несколько примеров. Примеры применения при этом подключают разные периферии в целях демонстрации возможностей микроконтроллеров семейства STM32W. Одновременно в одном приложении задействованы почти все периферии. Это безусловно демонстрирует возможности контроллера, но затрудняет понимание работы самой интересной периферии контроллера – радиочастотного модуля. Мы построили своё приложение на базе стандартного примера применения стека SimpleMac. При этом исключили практически всё лишнее и кода. Задача 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. Объём кода программы, в результате таких действий, упал с 30-ти до 10-ти кб.

Наиболее важные моменты этого проекта приводим в виде листинга.

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, провести инициализацию радиомодуля:

```
//инициализация радиомодуля
assert(ST_RadioInit(ST_RADIO_POWER_MODE_RX_ON)==ST_SUCCESS);
```

Установить адрес устройства и адрес сети (в нашем случае адрес 0x1604):

```
//установка адреса устройства в сети и адреса сети
ST_RadioSetNodeId(0x1604);
ST_RadioSetPanId(0x1604);
```

Выделить память под буферы приёма 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 приёмника (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);
```

Приём пакета. При успешном приёме 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.

```
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, которые так же можно наполнить своим кодом. Таким образом, можно создать приложение, которое будет весьма гибко реагировать на различные ситуации, связанные с отправкой и приёмом сообщений через эфир.

7. 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 очень высокой квалификации разработчика приложения. Интеграция радиочастотного модуля в приложение не является сложной задачей. Сравнить инициализацию беспроводного обмена с организацией обмена по UART можно тут: http://www.promelec.ru/stm/stm32/ в разделе «Примеры применения».

Архиважный вопрос для применения данной технологии – её цена. Как показано выше, цена этой технологии будет падать и через какое-то время станет соизмеримой с существующей ИК-технологией передачи данных.

Ссылки:

По вопросам применения обращайтесь в техническую поддержку компании Промэлектроника по адресу: support@promelec.ru

http://www.promelec.ru/stm/stm32/ - материал тренинга для STM32W (в разделе «примеры применения»)

http://www.st.com/internet/mcu/subclass/1377.jsp - обзор семейства контроллеров STM32W

http://www.st.com/stonline/stappl/resourceSelector/app?page=resourceSelector&doctype=DATASHEET&SubClassID=1377 – описания контроллеров STM32W

http://www.st.com/stonline/stappl/resourceSelector/app?page=resourceSelector&doctype=FIRMWARE&SubClassID=1377 – стеки и примеры применения для контроллеров STM32W

http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/USER_MANUAL/CD00262339.pdf - описание работы Simple Mac

http://supp.iar.com/Download/SW/?item=EWARM-KS32 – ссылка для скачивания бесплатной среды для программирования контроллеров STM32W