Миниатюрное цифровое переговорное устройство с радиусом действия до 300 метров

Сергей Гаевский, ведущий специалист ООО «Гамма Украина»
E-mail: info@nuvoton.com.ua

В статье представлен проект малопотребляющего беспроводного симплексного переговорного устройства, выполненного на базе микроконтроллера серии NUC505 от Nuvoton и радиомодуля RFM300W от HopeRF. Открытый исходный код проекта написан на языке C в среде Keil uVision 5. Подробно описывается техника программирования микроконтроллера для обработки звука в реальном времени и для синхронной демодуляции радиосигнала.
Статья может быть полезной разработчикам систем связи для стационарных и мобильных объектов или студентам в качестве основы для учебных проектов.

Речевое общение сегодня является неотъемлемой частью существования человека, без него немыслимо функционирование множества сервисов, в том числе обеспечивающих нашу безопасность. Мобильная связь прочно вошла во все сферы промышленности и быта, являясь базисом, на котором строится иерархия наших взаимоотношений. Использование публичной инфраструктуры, предоставляемой сотовыми операторами, накладывает определенные ограничения на доступность сервиса и выдвигает свои требования к клиентам. Так, отсутствует возможность моментальной связи между участниками группы, кроме того, обычно взимается абонентская плата или плата за предоставленное время разговора. Часто такие условия не являются оптимальными для пользователей и заставляют последних искать альтернативные способы голосовой связи, например, применять радиостанции. В отличие от сотовой связи, предоставляющей глобальное покрытие, радиостанции обеспечивают связь лишь в зоне обслуживания определенного радиуса с дополнительной возможностью организации интернет-шлюзов, например, для общения с удаленным диспетчером. Современные средства радиокоммуникаций среднего и малого радиуса действия сегодня широко используются для передачи потоковой мультимедийной информации. Подобные решения весьма эффективны в бизнесе (например, охрана в торговых залах), сфере досуга (коллективные виды спорта) и быта (домофоны, коммуникаторы в лифтах). Сдерживающими факторами являются сравнительно высокая стоимость абонентских устройств и необходимость их лицензирования. Поэтому актуальной является разработка дешевых и энергоэффективных цифровых устройств, функционирующих в ISM-диапазонах в пределах разрешенной мощности [1] и обеспечивающих передачу сжатого речевого сигнала в виде потока данных.

Как известно, оцифрованная речь является избыточной в плане информации, действительно необходимой для ее понимания. «Лишнюю» информацию можно исключить, разбивая речь на небольшие фрагменты и заменяя их описателями, содержащими характеристики фрагмента, в той или иной степени достаточные для его восстановления в виде, пригодном для понимания человеком. Существуют разнообразные алгоритмы кодирования, оптимальные для различных применений. Например, GSM-FR кодек в системах сотовой связи 2G-поколения обеспечивает десятикратное сжатие речи, в то время как кодек MELPE-600, используемый в спутниковых системах связи военного назначения, сжимает речь более чем в 200 раз. Степень сжатия речи определяет необходимую скорость битового потока, передаваемого по радио и, соответственно, так называемый показатель бюджета канала, пропорционально влияющий на дальность связи при ограниченной мощности передатчика. Обратной «стороной медали» является некоторая потеря индивидуальных особенностей речи и общей разборчивости при высоких степенях сжатия. Дополнительным преимуществом цифровой передачи является возможность сильной криптографической защиты речевой информации, недоступной в аналоговых радиостанциях. Так, 128-битный уровень защиты легко достигается при использовании современных потоковых шифров и позволяет использовать как индивидуальные, так и групповые ключи шифрования. Также, в случае необходимости, имеется возможность расширения защиты индивидуального сеанса связи за счет так называемой совершенной обратной секретности (невозможности расшифровать ранее записанный радиообмен, даже получив полный доступ к абонентским терминалам и их содержимому). Данная функция предполагает обмен публичными ключами и вычисление временного ключа шифрования в начале каждого сеанса связи, что легко реализуемо при использовании цифровой передачи данных.

Целью данного проекта являлась разработка миниатюрного и недорогого беспроводного переговорного устройства (радиостанции), не требующего лицензирования и обеспечивающего симплексную связь с качеством речи, сопоставимым с таковым в сотовой телефонии. Дополнительным требованием было снижение тока потребления в режиме дежурного приема до значений, позволяющих устройству работать в течение нескольких месяцев от одной зарядки аккумулятора. Для реализации проекта были выбраны доступные бюджетные радиокомпоненты: микроконтроллер Nuvoton **NUC505DSA** и радиомодуль HopeRF **RFM300W**. Необходимая обвязка включала стабилизатор питания 3.3 В с током до 100 мА, кварцевый резонатор 12 МГц, микрофон, наушники (или динамик с отключаемым усилителем мощности класса D на микросхеме PAM8403), кнопку тангенты и светодиод, а также несколько блокировочных конденсаторов и резисторов. Устройство питалось от Lilo-аккумулятора емкостью 2400 мА·ч или сменного комплекта батарей с напряжением не менее 3.2 В.

Радиомодуль RFM300W от компании HopeRF (рис. 1), базирующийся на FSK-трансивере **CMT2300A**, неоднократно [2] описывался в предыдущих статьях [3, 4]. Он представляет собой цифровой приемопередатчик, интегрирующий цепи генератора опорной частоты, входного усилителя с АРУ, FSK/OOK модема с АПЧ, выходного усилителя мощностью до 100 мВт, модуля пакетирования с буферами для передаваемых и принимаемых данных, модуля синхронизации и модуля таймеров для автоматической смены режимов работы в цикле (рис. 2). Трансивер поддерживает как пакетный режим работы, подробно описанный в нашей предыдущей статье [4], так и «прозрачный» режим, использовавшийся в данном проекте. Особенностью «прозрачного» режима работы является отсутствие предварительной обработки данных в трансивере, осуществляющем лишь частотную модуляцию и демодуляцию радиосигнала с автоматической подстройкой частоты приемника.

Сами данные для передачи полностью формируются микроконтроллером: он определяет битрейт и подает поток битов на вход трансивера. На приемной стороне трансивер формирует соответствующий поток принятых «сырых» битов на выходе. Существует возможность определения битрейта потока принимаемых битов и сопровождения выдачи данных импульсами тактирования (так называемый механизм CDR, встроенный в трансивер), но мы предпочли программную обработку «сырых» данных средствами микроконтроллера для большей наглядности и познавательности проекта.

Предварительная настройка радиочастотных параметров производилась с помощью графической утилиты RFPDK от компании HopeRF [5], позволяющей сформировать банк значений всех регистров трансивера для его инициализации на старте. Были созданы отдельные настроечные блоки для работы трансивера в обычном режиме и в режиме дежурного приема. Смена этих режимов осуществлялась путем полного перезапуска и повторной инициализации трансивера, на что требовалось около 50 мс.

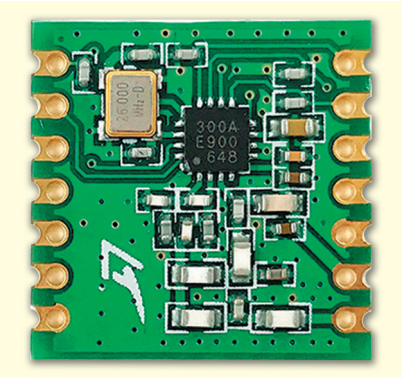

Рис. 1. Модуль RFM300 на базе трансивера CMT2300A с цепями обвязки

При настройке трансивера в рабочем режиме на основной вкладке программы RFPDK выбирались базовая рабочая частота (Frequency = 868.00 МГц), скорость передачи данных (Data rate = 5.0 кбит/с), тип модуляции (Modulation = FSK), девиация частоты (Deviation = 5 кГц) и мощность передатчика (TX Power = +13 дБм), что соответствовало 20 мВт. Остальные настройки устанавливались по умолчанию.

На вкладке FSK **Demod Settings** подстройка битрейта была отключена (CDR = None). Таким образом, в режиме приема трансивер выдавал «сы-

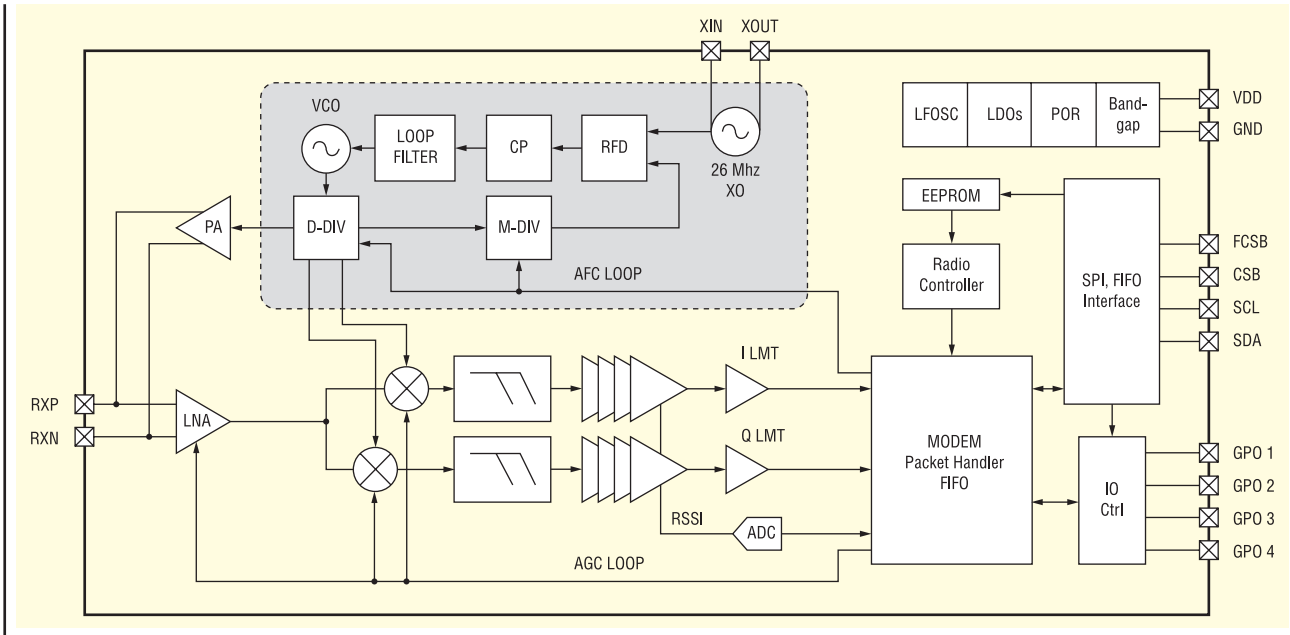

Рис. 2. Блок-схема трансивера CMT2300A

рые» данные, а функция синхронизации возлагалась на микроконтроллер. На вкладке **Baseband 1 Settings** выбирался «прозрачный» режим работы (Data Mode = Direct). На вкладке **Feature Settings** функция подавления выхода приемника при отсутствии сигнала была отключена (Dout Mute = Off). Дополнительно на этой же вкладке был установлен минимальный уровень детектора снижения напряжения питания (LBD Trashold = 1.8 В), что предотвращало перманентную блокировку трансивера при кратковременном падении напряжения, требующую полного перезапуска устройства.

Для формирования альтернативного набора параметров, обеспечивающих режим дежурного приема, были произведены дополнительные настройки. На вкладке **Operation settings** были активированы таймер приема (RX Duty-Cycle = On, RX Timer = On, RX Time T1 = 20.0 мс, RX Exit State = SLEEP) и таймер сна (Sleep Timer = On, Sleep Time = 500.0 мс). Для быстрой смены режимов приема и сна была отключена калибровка тактовой частоты таймеров (LFOSC Calibration = Off). Это в некоторой степени снижало точность отсчета интервала сна, что не было критичным для данного приложения. Кроме того, на этой же вкладке были заданы параметры обнаружения несущей, необходимые для разблокирования выхода приемника в случае наличия передачи в радиоканале (RSSI Valid Source = PJD и PJD Window = 8 Jumps). Выбор этих параметров определял метод обнаружения передачи: вместо замера силы сигнала детектировались скачки фазы, характерные для частотной модуляции при смене значения передаваемых битов. Сигнал считался обнаруженным при детектировании 8 скачков фазы (прием байта 0xAA или 0x55). При этом существенно снижалась вероятность ложных срабатываний от помех и сигналов других радиосистем, т. к. обнаружение полезного сигнала было привязано к используемому нами битрейту 5 кбит/с. Надежность обнаружения сигнала обеспечивалась соответствующей маской, накладываемой нашим передатчиком на передаваемые данные. При наличии речевых пауз, фиксируемых детектором голоса (VAD), кодек формировал блок из 12 нулевых байт, которые после маскировки модулировались как последовательность байт 0x55 или 0xAA и определялись приемником как преамбула.

На вкладке **Feature Settings** был включен режим подавления выхода приемника (Dout Mute = On) и настроен режим детектирования уровня принимаемого сигнала (RSSI Detect Mode = Always, RSSi Filter Settings = 8-tap, RSSI Offset = 10, RSSI Offset Sign = 0). Благодаря вышеуказанным настройкам трансивер после соответствующей первичной инициализации входил в состояние дежурного приема и не требовал дополнительного управления со стороны микроконтроллера. В этом режиме трансивер слушал эфир в течение 20 мсек, потребляя 8 мА тока, затем на 500 мс погружался в сон, потребляя при этом 800 нА. Таким образом, среднее энергопотребление составляло около 320 мкА с учетом периода выхода из режима сна с ожиданием запуска опорного генератора радиочастоты. Выход приемника находился в стабильном нулевом состоянии до обнаружения частотно-модулированного сигнала в радиоканале, после чего трансивер выдавал биты преамбулы в течение периода активного приема. Микроконтроллер использовал прерывание по фронту этого сигнала для выхода из режима глубокого сна, работая при этом на пониженной тактовой частоте. В течение секунды производился анализ состояния выхода приемника. При наличии постоянного битового потока микроконтроллер аппаратно перезапускался, производил сброс трансивера и перенастраивал его для функционирования в рабочем режиме. В случае отсутствия постоянного битового потока микроконтроллер возвращался в режим глубокого сна, считая, что прерывание было вызвано радиопомехой или спонтанным пакетом чужой радиосистемы. При тестировании в условиях города, подобные спонтанные прерывания возникали не чаще одного раза в несколько минут, поэтому их влияние на общее энергопотребление в режиме дежурного приема было незначительным.

После формирования утилитой RFPDK выходных текстовых файлов с расширением **.exp* отдельно для рабочего режима и режима дежурного приема, они конвертировались с помощью утилиты *cmt2300a_params_convert.py* в соответствующие заголовочные h-файлы и подключались к исходному коду проекта.

Критериями выбора микроконтроллера являлись наличие ядра Cortex M4F с аппаратной поддержкой математических вычислений с плавающей точкой, тактовая частотой процессора не ниже 72 МГц, объем постоянной Flash-памяти не менее 128 Кбайт и оперативной памяти не менее 32 Кбайт. Наиболее бюджетным выбором, соответствующим вышеуказанным характеристикам, явились микроконтроллеры Nuvoton **M451LG6AE** [6] и **NUC505DSA** [7]. Блок-схемы данных микроконтроллеров представлены на рисунках 3 и 4.

Последний вариант выгодно отличался наличием встроенного аудиокодека, исключающего необходимость использования внешнего микрофонного усилителя и выходного усилителя мощности для наушников, необходимых при записи звука с помощью АЦП и воспроизведении с помощью ЦАП (вариант подобной обработки звука был описан в нашей статье [8]).

Серия микроконтроллеров Nuvoton NUC505, представленная нами ранее [9], использует встроенную Flash-память, подключенную к ядру по SPI-интерфейсу, что существенно снижает ее стоимость. Недостатком подобного решения является чрезвычайно низкая скорость исполнения кода непосредственно из Flash-памяти. Так, у микроконтроллеров серии NUC505 при максимальной тактовой частоте ядра Cortex M4F, равной 100 МГц, обеспечивается реальное быстродействие всего в несколько MIPS, что критически мало для сжатия речи. Единственным выходом является использование загрузчика, обычно применяемого микропроцессорами: на старте содержимое Flash-памяти, включая исполняемый код и постоянные табличные данные, копируется в ОЗУ, а затем с помощью аппаратных средств область адресов Flash-памяти, начиная со стартового вектора, отображается на адреса ОЗУ. Код в ОЗУ выполняется без задержки на чтение, что в описанном случае обеспечивает быстродействие около 125 MIPS, вполне достаточное для поставленных нами задач. Единственным ограничивающим фактором является размер ОЗУ: в выбранном микроконтроллере он составляет 128 Кбайт. В него должны поместиться исполняемый код (секция CODE), табличные данные (секция RO), пользовательские переменные и массивы, область стека и область «кучи», необходимая для динамического выделения памяти (секция ZI).

Nuvoton предоставляет шаблоны проектов с уже готовыми scatter-файлами для процесса сборки с размещением исполняемого кода полностью в ПЗУ (по умолчанию), полностью в ОЗУ или частично в ОЗУ (только код, требующий высокого быстродействия). Также имеется шаблон для организации динамической загрузки необходимых

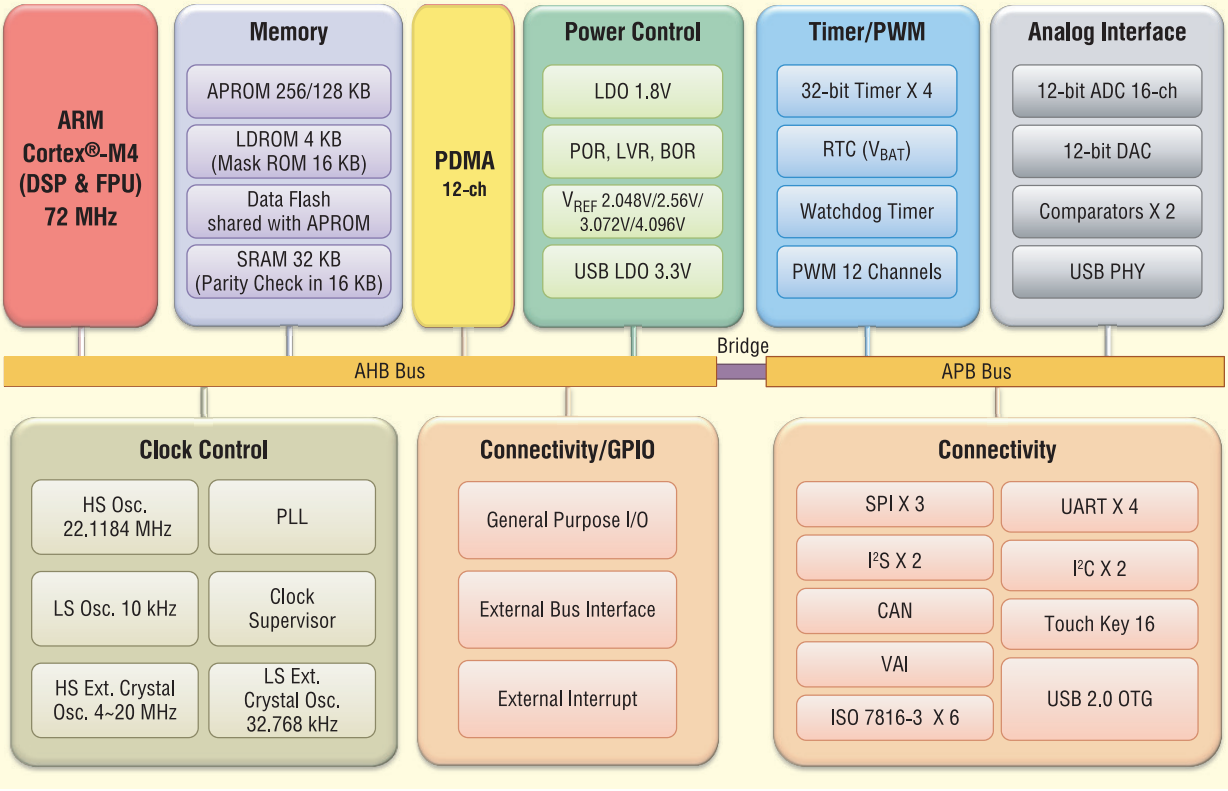

Рис. 3. **Блок-схема микроконтроллера M451LG6AE**

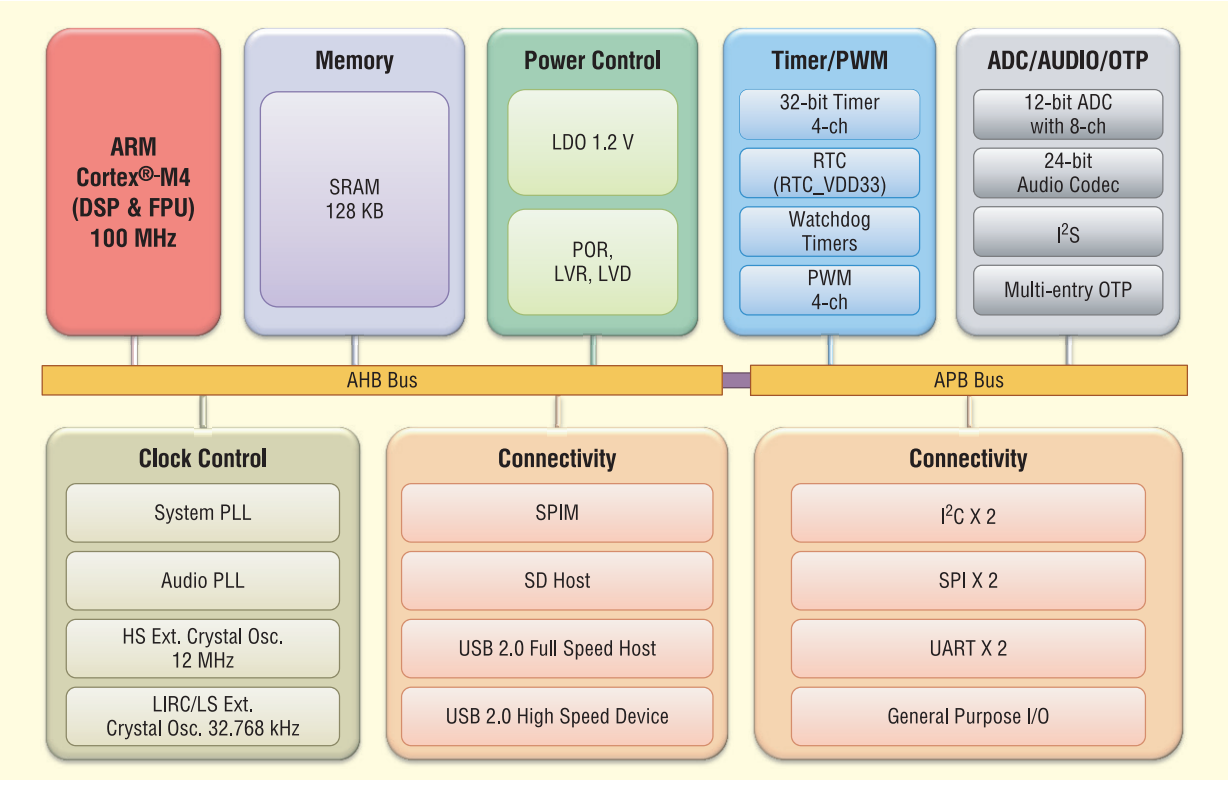

Рис. 4. **Блок-схема микроконтроллера NUC505DSA**

фрагментов кода в ОЗУ во время исполнения программы (механизм оверлеев). Для упрощения нашего проекта мы выбрали вариант размещения всего исполняемого кода в ОЗУ, за исключением служебных процедур инициализации микроконтроллера и самого загрузчика, помещаемых в ПЗУ. Основной объем кода формировался за счет AMR-аудиокодека и его табличных данных. Для максимального уменьшения этого объема из кодека была удале-

на поддержка режимов сжатия, кроме единственного используемого нами режима AMR-4750. В имеющемся у нас исходном коде AMR-кодека [10] кодер и декодер полностью независимы друг от друга и используют различную математику: кодер — с плавающей точкой, а декодер — 32-битную целочисленную. Поэтому табличные данные также независимы: хотя и содержат одинаковые значения, но в различном представлении. При уровне оптимизации O2 (с максимальной скоростью исполнения) размер кода AMR-кодека составил 35492 байта (из них 10796 — кодер, 24696 байт — декодер), размер табличных данных — 27697 байт (из них 19140 байт — кодер, 8556 байт — декодер), размер области динамически выделяемой ОЗУ — 6592 байта (из них 5316 байт — кодер, 1276 байт — декодер). Таким образом, полный AMR-кодек занимает около 64 Кбайт ПЗУ и дополнительно требует еще около 7 Кбайт динамически выделяемой памяти. Для обеспечения стабильной работы кодека в нашем проекте было зарезервировано 12 Кбайт ОЗУ для «кучи» и 8 Кбайт— для стека. При рабочей частоте Cortex M4F процессора 100 МГц и выполнении кода в ОЗУ время процедуры кодирования составило около 8.7 мс в случае наличия голоса и около 2.9 мс в случае его отсутствия. Время декодирования составило около 1.5 мс, что суммарно с длительностью процедуры кодирования в случае дуплексной обработки равнялось половине длительности обрабатываемого голосового фрейма продолжительностью 20 мс. Исходя из приведенных расчетов, можно сделать вывод, что в данных условиях около половины вычислительных ресурсов процессора остается свободным и может быть задействовано на усмотрение пользователя, например, для обеспечения стойкого шифрования речи с использованием самых надежных криптографических алгоритмов.

Исходный код проекта хорошо структурирован и включает файлы AMR-кодека (папка *flt*), файлы управления аппаратными модулями (папка *hw*), файлы управления трансивером (папка *rf*) и файлы модема (папка *mdm*). В корневом каталоге находится основной файл *main.c* и файл обработчика аудиоданных *prc.c*. Основной файл содержит вызовы процедур стартовой инициализации и основной рабочий цикл программы, в котором обеспечивается взаимодействие периферийных модулей и обработчика аудиоданных одновременно для передачи и приема (дуплекс), хотя трансивер может работать только в одном из этих режимов. Перенастройка трансивера, осуществляемая через SPI-подобный интерфейс, является блокирующей процедурой и выполняется только при смене режима работы «прием – передача» на входе в основной цикл программы.

Обработчик аудиоданных, представленный в файле *prc.c*, содержит две FIFO-очереди (одна — для передачи, другая — для приема), каждая из которых состоит из трех сжатых речевых фреймов. Речевой фрейм, обрабатываемый AMR-кодеком, — это 160 16-битных PCM сэмплов с частотой сэмплирования 8 кГц. Длительность такого фрейма составляет 20 мс. В сжатом виде фрейм представляет собой 12-байтный блок данных (один бит — индикатор голоса или тишины, 95 бит — описатель голоса или нули в случае тишины). Определение речевых пауз производится голосовым детектором, встроенным в AMR-кодек. Таким образом, во время голосовых пауз существует возможность передачи произвольных пользовательских данных, например, для синхронизации однонаправленных счетчиков фреймов или обмена сесионными публичными ключами при использовании шифрования.

Интерфейс обработчика аудиоданных в режиме передачи обеспечивает:

- сжатие речевого фрейма, полученного от записывающего аудиоустройства и помещение полученного блока данных в очередь;
- извлечение блока данных из очереди и модуляция его в последовательность бит для отправки на вход трансивера.

В режиме приема обеспечивается:

- демодуляция последовательности бит, захваченных с выхода трансивера и помещение полученного блока данных в очередь;
- извлечение блока данных из очереди, декодирование и выдача PCM фрейма для дальнейшего проигрывания через устройство воспроизведения аудио.

Дополнительно реализован механизм, позволяющий избежать переполнения или недонаполнения очереди приемника. Темп проигрывания аудиоблоков определяется частотой кварцевого резонатора приемника, в то время как темп их поступления — частотой кварцевого резонатора удаленного передатчика. Допустимая погрешность кварцевых резонаторов может достигать 20 ppm, таким образом, разность частот в худшем случае составляет 40 ppm. Для компенсации такой разности предусмотрен механизм контроля наполнения очереди приемника и формирование сигнала ускорения или замедления темпа проигрывания аудио. При использовании проигрывания аудио с помощью ЦАП его скорость может быть скорректирована аппаратно подстройкой соответствующего таймера, но при использовании встроенного кодека скорость проигрывания равна скорости записи. Поэтому любое изменение частоты сэмплирования приведет к нарушению скорости передачи, что недопустимо в дуплексном режиме. Единственным выходом является программное изменение в небольших пределах частоты сэмплирования аудиопотока перед его выдачей на проигрыватель. В нашем проекте это обеспечивал ресемплер, вычисляющий значение выдаваемого сэмпла по двум соседним сэмплам методом линейной интерполяции. Ухудшения качества речи на выходе такого ресемплера вполне приемлемо для телефонии, а вычислительные затраты намного ниже по сравнению с более сложными алгоритмами, используемыми, например, для обработки музыки. Незначительное изменение тональности речи в пределах диапазона подстройки незаметно на слух, что избавляет от необходимости применения более сложных алгоритмов изменения темпа речи при сохранениии ее тональности. Кроме того, процесс ресемплирования запускается только при проигрывании непрерывных речевых фрагментов длиной в несколько десятков секунд. При наличии речевых пауз наполнение очереди корректируется путем вставки или пропуска фреймов тишины, при этом наблюдается незначительное увеличение или уменьшения пауз между словами, что также незаметно на слух.

Реализация аппаратного захвата и проигрывания аудио описана в файле *aud.c* и зависит от возможностей конкретного микроконтроллера. В случае выбора серии Nuvoton NUC505 используется встроенный кодек, предварительно настраиваемый на этапе инициализации, использующий прямой доступ к памяти (DMA) для записи фрейма в буфер и формирующий аппаратное прерывание при его готовности. Для обеспечения одновременного непрерывного процесса записи и воспроизведения в памяти выделяется два двойных буфера: в то время, когда кодек работает с одной половиной буфера в режиме DMA, а вторая половина доступна микропроцессору.

Интерфейс программного аудиомодуля представлен двумя неблокирующими функциями опроса готовности буферов записи и воспроизведения, постоянно вызываемыми из основного цикла программы. Каждые 20 мс при готовности аудио функция записи возвращает указатель на буфер, содержащий записанный фрейм, а функция воспроизведения — на буфер, куда необходимо поместить новый фрейм для дальнейшего его проигрывания. При получении ненулевого указателя необходимо вызвать соответствующую функцию модуля обработки аудио для помещения записанного фрейма в очередь передатчика или извлечения следующего фрейма для его проигрывания из очереди приемника.

Реализация выдачи битового потока на трансивер в режиме передачи и захвата данных с трансивера в режиме приема описана в файле *spp.c*. Для обеспечения дуплексной работы используются два независимых аппаратных SPI модуля микроконтроллера. Оба модуля работают в режиме мастера, их тактовая частота равна 16-кратному битрейту радиоканала и составляет 80 кГц. Таким образом, приемник в течение одного бита данных захватывает 16 отсчетов, а в передатчике каждый бит кодируется записью значения 0x0000 или 0xFFFF в регистр данных SPI.

Использование приемником и передатчиком двух различных аппаратных SPI-модулей необходимо для обеспечения независимой подстройки тактовой частоты приемника, которая должна соответствовать тактовой частоте передатчика на удаленной стороне. Как уже говорилось выше, разница в частоте кварцевых резонаторов передатчика и приемника может достигать 40 ppm, и в этих пределах частота тактирования SPI приемника должна быть скорректирована для обеспечения синхронизации и непрерывности потока принимаемых данных. Корректирующий сигнал формируется демодулятором (см. ниже) и на короткое время изменяет коэффициент делителя тактового сигнала аппаратного модуля SPI приемника, удерживая его среднюю частоту в пределах частоты удаленного передатчика. При этом в дуплексном режиме частота аппаратного модуля SPI передатчика должна оставаться стабильной в качестве мастер-сигнала для синхронизации удаленного приемника.

Модули SPI микроконтроллеров серии NUC505, к сожалению, не имеют прямого доступа к памяти, поэтому для работы с ними мы применяли прерывания. Использование 32-битного режима работы и встроенного аппаратного 8-уровневого буфера обеспечивают обработку всего 6–7 прерываний в течение длительности аудиофрейма (20 мс), что незначительно сказывается на быстродействии в целом. Интерфейс программного модуля данных аналогичен аудиоинтерфейсу и представлен двумя неблокирующими функциями, возвращающими ненулевые указатели на соответствующие половины двойных буферов данных передатчика и приемника при их готовности.

Битовый поток, захваченный с выхода приемника, поступает на демодулятор, описанный в файле *mdm.c*. Основными задачами демодулятора являются:

- определение границ информационного бита, представленного 16 отсчетами (двумя байтами) и его детектирование — определение значения 0 или 1, а также их вероятности (LLR);
- определение границ блоков данных, соответствующих аудиофреймам, упаковка принятых информационных бит в 12-байтный блок;
- строгая синхронизация счетчика принятых блоков со счетчиком на стороне передатчика (необходима для криптографических целей);
- формирование флага обнаружения или потери синхронизации для дальнейшего управления работой приемника (включением выходного усилителя и переключением в режим дежурного приема).

Первая из поставленных задач заменяет аппаратный механизм CDR, отключенный нами в трансивере. Так как каждый бит представлен 16-ю отсчетами, то предполагается, что эти отсчеты в своем большинстве будут одинаковы на протяжении бита, за исключением, возможно, небольшой области смены значения бита на противоположное и ошибок вследствие действия радиопомех. Таким образом, демодулятор подсчитывает количество совпадающих отсчетов в пределах границы бита для каждой из 16 возможных позиций. Результат усредняется с помощью фильтра на протяжении нескольких обработанных блоков данных, затем определяется позиция с наибольшим количеством совпадений. Исходя из выбранной позиции, анализируется 8 отсчетов, расположенных в центре предполагаемой области информационного бита, определяется значение бита («0» или «1») и его вероятность по количеству совпадающих отсчетов. Сигнал для подстройки битрейта приемника формируется таким образом, чтобы удерживать битовую границу в пределах седьмого-восьмого отсчета. При небольшой разности битрейта передатчика и приемника наблюдается медленный уход границы бита в одну или другую сторону, соответственно, периодически формируется сигнал ошибки, корректирующий частоту тактирования приемного SPI-модуля и удерживающий битовую границу в указанных пределах.

Для определения границы блока используется механизм синхронизации демодулятора «на лету». Обычно в случае пакетной передачи данных аналогичная задача решается отправкой преамбулы с последующей синхронизирующей последовательностью в начале каждого пакета. Подобный механизм добавляет к полезным данным значительную избыточность. Для непрерывной потоковой передачи оптимальным является вставка короткой синхронизирующей битовой последовательности между блоками данных и использование фильтрации для накоплении информации о ее позиции в течение приема нескольких блоков. Мы использовали 4-битовую синхронизирующую вставку в виде увеличивающегося счетчика блоков, добавляя ее в поток данных после каждых 96 бит полезной нагрузки (12 байт, соответствующих сжатому аудиофрейму длительностью 20 мс). Таким образом, избыточность составила всего 4%, и в течение 20 мс по радиоканалу передавалось 100 бит данных, что определяло рабочий битрейт в 5 Кбит/с. Поиск границы фрейма производился путем подсчета совпадений текущего значения синхровставки (счетчика блоков) с предыдущим, увеличенным на единицу. Соответствующее количество совпадений накапливалось для каждой из 100 возможных битовых позиций и определялась позиция с максимальным количеством таких совпадений, предположительно являющаяся границей блока данных. Зная границу блока данных, демодулятор упаковывал принимаемые биты в блок, расставляя каждый из них на свое место.

Значения синхровставок (номера блоков данных) использовались для синхронизации внутреннего счетчика принятых блоков. Синхронизация выполнялась с использованием фильтрации на протяжении нескольких последовательных блоков, что исключало спонтанные ошибки при наличии помех и формировало стабильное значение номера принятого блока, пригодное для использования в криптографических це-

лях (для потокового шифрования речи). Для определения флага состояния синхронизации приемника, указывающего на наличие сигнала в радиоканале, использовались счетчики совпадений и несовпадений принятого номера блока с предполагаемым его значением. При достижении определенных порогов состояние флага изменялось, динамически отображая наличие или отсутствие синхронизации демодулятора.

В файле *com.c* описан интерфейс работы с аппаратным модулем UART, обеспечивающим прием пользовательских команд и передачу тестовой информации о состоянии устройства. Микроконтроллеры серии NUC505 имеют 64-байтные FIFO-буферы UART, что гарантирует передачу и прием пакетов данных соответствующей длины без использования DMA или прерываний. Готовность принятого пакета определяется по аппаратному флагу тайм-аута, который выставляется при неактивности приемника UART в течение заданного интервала времени длительностью в 100 бит (около 1 мс при скорости обмена 115200 бит/с) и наличии данных в аппаратном буфере. Принятые данные помещаются в 64-байтный массив, и интерфейсная неблокирующая функция, постоянно вызываемая из основного цикла, возвращает их длину.

В данном проекте возможность смены частотного канала была реализована с помощью команды:

F=xxx<CR><LF>,

где xxx — номер канала от 0 до 255.

Фактическая частота определяется увеличением базовой частоты нулевого канала (в нашем случае равной 868.00 МГц) на значение, равное номеру канала, умноженному на интервал (в нашем случае установленный в 10 кГц). Таким образом, частота канала 255 равна 870.55 МГц. Новая частота канала устанавливается при смене режима работы с приема на передачу и наоборот. При выборе канала пользователь должен следить за тем, чтобы оставаться в пределах ISM-диапазона, разрешенного местным законодательством в сфере планирования радиочастотного спектра.

В режиме приема при отсутствии активности пользователя (переключении в режим передачи нажатием кнопки тангенты) и активности в радиоканале в течение установленного промежутка времени (в нашем проекте 60 с) коммуникатор переходит в режим дежурного приема с минимальным энергопотреблением. Работа в этом режиме описана в файле *pdc.c*. Трансивер переводится в режим автоматического чередования приема и сна путем полного перезапуска и загрузки альтернативного комплекта радиопараметров в его регистры. Затем микроконтроллер останавливает все работающие аппаратные модули, потребляющие энергию, понижает рабочую частоту шины процессора до минимально возможного значения (для серии NUC505 это 750 кГц), активирует прерывания по изменению состояния выхода трансивера и кнопки тангенты и входит в состояние глубокого сна. К сожалению, архитектура серии NUC505 не предназначена для эффективного энергосбережения, так как требуется непрерывная регенерация данных в ОЗУ, при этом потребляется ток около 700 мкА, что в сумме со средним потреблением трансивера в режиме периодического приема (см. выше) составляет около 1 мА. При использовании серии M451 в режиме глубокого сна удается снизить потребляемый микроконтроллером ток до 20 мкА и таким образом почти втрое продлить время работы устройства от батареи в режиме дежурного приема.

После выхода из режима глубокого сна по обнаружению активности в радиоканале микроконтроллер в течение заданного промежутка времени (1 с) периодически сканирует выход трансивера и подсчитывает количество случаев определения логической единицы. После сравнения результата с пороговым значением, принимается решение о переходе в рабочий режим или возвращении обратно в режим сна. Этот механизм предотвращает выход из режима дежурного приема при обнаружении единичного «чужого» пакета, тем самым дополнительно экономя энергию батареи.

Переключение из режима дежурного приема в рабочий режим происходит путем полного перезапуска микроконтроллера и повторной инициализации регистров трансивера. К сожалению, в микроконтроллере серии NUC505 при выполнении кода в ОЗУ не удалось с помощью программного сброса выполнить полноценную передачу управления коду загрузчика, расположенному в Flash-памяти, поэтому мы использовали аппаратный сброс путем установки нулевого уровня на выводе микроконтроллера, физически соединенном с цепью сброса. Для микроконтроллеров серии M451, выполняющих

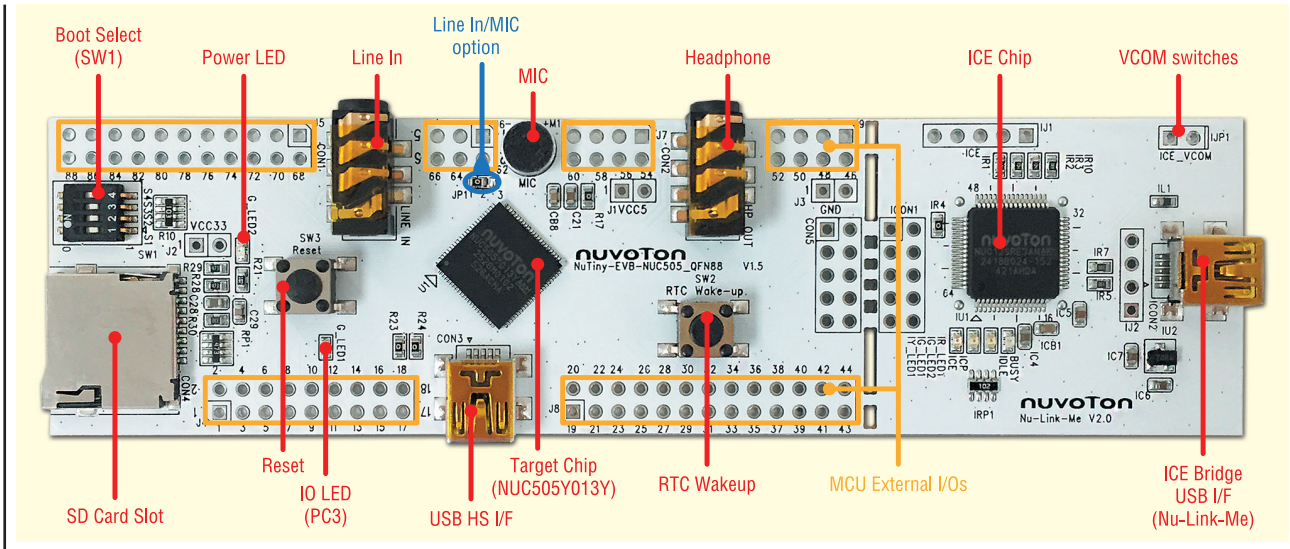

Рис. 5. Отладочная плата NuTiny-SDK-NUC505 с микроконтроллером NUC505YO13Y

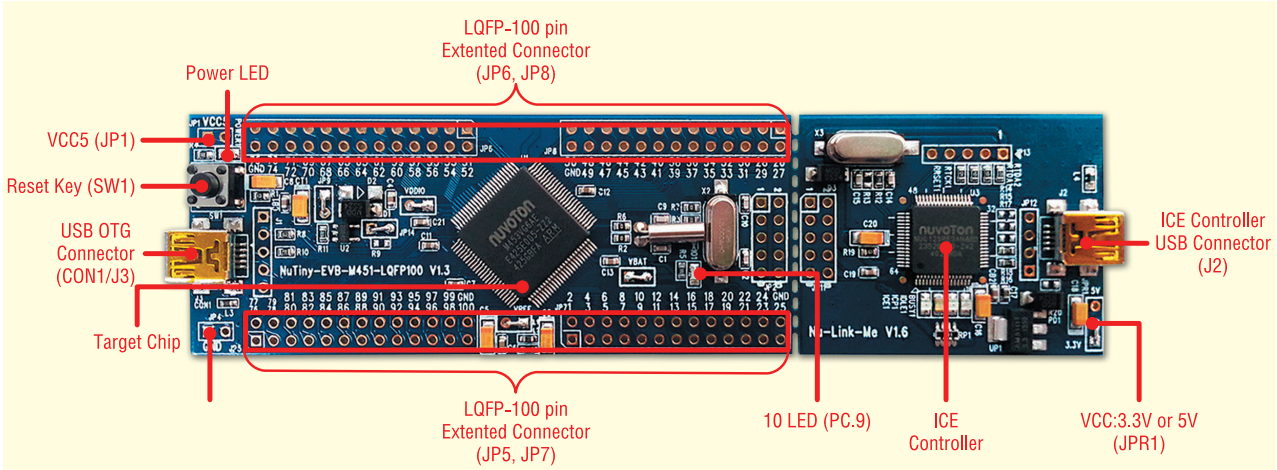

Рис. 6. Отладочная плата NuTiny-SDK-M453 с микроконтроллером M453VG6AE

код непосредственно из Flash-памяти, программный сброс приводит к полному перезапуску, что избавляет от необходимости использования дополнительной цепи аппаратного сброса.

Нужно отметить, что после начала передачи в радиоканале устройству, находящемуся в режиме дежурного приема, потребуется около 3 с до включения усилителя динамиков и выдачи голосового сообщения. При этом выполняются следующие операции: открытие окна приема (0–500 мс), определение активности в канале по последовательным скачкам фазы сигнала (1–2 мс), дополнительный анализ выхода приемника (1 с), аппаратный сброс с переносом исполняемого кода в ОЗУ (около 300 мсек), инициализация трансивера (50 мс) и синхронизация демодулятора (около 200 мс).

В случае работы трансивера в режиме обычного приема задержка после включения передатчика составляет менее 200 мс и определяется временем, необходимым на синхронизацию демодулятора. Светодиодный индикатор отображает текущее состояние устройства: в режиме передачи он зажигается при наличии речи, в режиме приема при отсутствии сигнала вспыхивает раз в секунду, а при его наличии быстро мигает. В режиме дежурного приема светодиод для экономии заряда батареи не включается.

На этапе разработки проекта [11] использовалась отладочная плата **NuTiny-SDK-NUC505** [12] с установленным на ней старшим представителем серии NUC505: микроконтроллером **NUC505YO13Y** в 88-выводном корпусе (рис. 5). Шаблон нашего проекта соответствовал данному микроконтроллеру. При переносе проекта на один из младших представителей серии (например, на рекомендуемый нами микроконтроллер **NUC505DSA** необходимо обратить внимание на наличие физических портов ввода-вывода, использующихся в проекте, и при необходимости переназначить отсутствующие сигналы в файлах *pdc.c*, *gpc.c* и *gpc.h*. В качестве альтернативного решения нами представлен параллельный проект [13] для серии M451, использующий вместо аудиокодека модули АЦП и ЦАП для записи и проигрывания аудио. Проект был реализован на отладочной плате **NuTiny-SDK-M453** [14] с установленным на ней микроконтроллером **M453VG6AE**, но с вышеприведенными оговорками совместим с более дешевыми младшими представителями серии, например **M451LG6AE** (рис. 6). Подключение к данным отладочным платам модуля **RFM300W** и других необходимых внешних элементов показано в таблицах 1 и 2 соответственно.

Испытания устройства проводились в условиях умеренной городской застройки, устройства находились в руках операторов. В качестве антенн использовались четвертьволновые отрезки провода длиной 85 мм. Связь обеспечивалась на расстоянии около 300 метров. Сигнал успешно преодо-

Таблица 1. Подключение внешних модулей к отладочной плате NuTiny-SDK-NUC505

Порт МС	Вывод платы	Сигнал	Примечание	Назначение
Подключение модуля трансивера RFM300				
PB12	80	SPI1 MOSI	Подключено через резистор 1 кОм	GPIO1
PB5	46	SPI0 MISO		
PB0	41	INT	Прерывание по активности приемника	
PB6	47	IN/OUT	Данные управления трансивером	SDIO
PB7	48	OUT	Такт управления трансивером	SCLK
PB8	50	OUT	Сигнал выбора трансивера	CSB
PB9	51	OUT	Доступ к буферу данных трансивера	FCSB
VDD	52	PWR	Питание трансивера 3.3 В	VDD
GND	89	PWR		GND
Дополнительные цепи управления				
PA10	30	OUT	Управление внешним УНЧ	1 — вкл. УНЧ
PA11	31	OUT	Управление внешним микрофоном	1 — вкл. МК
PC0	7	OUT	Сигнал аппаратного сброса	Соединить
RST	1	RST IN	Цепь сброса микроконтроллера	
Интерфейс пользователя				
PA9	29	UART1 RX	Посылка команды смены радиоканала	115200 бит/с
PA8	28	UART1 TX	Выдача тестового лога	115200 бит/с
PC12	83	IN, pull up	Тангента, кнопка замыкает на GND	Соединить, на кнопку
PB1	42	INT	Соединить с тангентой, прерывание	
PC13	84	OUT	Светодиод, анод соединить с VCC ч/з R = 1 кОм	
PC6	14	IN, pull up	Джампер включения режима тестовой передачи (непрерывный счет)	Соединить для включения теста
PC5	13	OUT	Земля для джампера	

Таблица 2. Подключение внешних модулей к отладочной плате NuTiny-SDK-M453

Порт МС	Вывод платы	Сигнал	Примечание	Назначение
Подключение модуля трансивера RFM300				
PA5	61	SPI1 MOSI	Подключено через резистор 1 кОм	GPIO1
PE10	68	SPI0 MISO		
PE5	54	INT	Прерывание по активности приемника	
PE1	65	IN/OUT	Данные управления трансивером	SDIO
PE8	66	OUT	Такт управления трансивером	SCLK
PE9	67	OUT	Сигнал выбора трансивера	CSB
PE4	53	OUT	Доступ к буферу данных трансивера	FCSB
VDD	64	PWR	Питание трансивера 3.3 В	VDD
GND	63	PWR		GND
Дополнительные цепи управления				
PA14	85	OUT	Управление внешним УНЧ	1 — вкл. УНЧ
PA15	86	OUT	Управление внешним микрофоном	1 — вкл. МК
Интерфейс пользователя				
PD6	21	UART0 RX	Посылка команды смены радиоканала	115200 бит/с
PD1	15	UART0 TX	Выдача тестового лога	115200 бит/с
PB1	92	ADC IN	Микрофонный вход, DC = 1.5 В, 1 В эфф.	На вход УНЧ
PB0	91	DAC OUT	НЧ выход, 1 В эфф.	На выход МК
PC7	52	IN, pull up	Тангента, кнопка замыкает на GND	На кнопку
PC6	51	OUT	Светодиод, анод соединить с VCC ч/з R = 1 кОм	На светодиод
PB13	1	IN, pull up	Джампер включения режима тестовой передачи (непрерывный счет)	Соединить для включения теста
PB14	2	OUT	Земля для джампера	

левал препятствие в виде железобетонного здания, позволяя организовать уверенную связь в пределах нескольких этажей. В режиме активного приема общий потребляемый ток составлял около 50 мА, в режиме передачи при выходной мощности трансивера 20 мВт — около 90 мА.

Таким образом, разработанные нами проекты демонстрируют возможность использования Cortex M4F микроконтроллеров общего назначения **Nuvoton NUC505DSA** (со встроенным аудиокодеком) или **Nuvoton M451LG6AE** (со встроенными 12-битными модулями АЦП и ЦАП) вместе с радиотрансивером **HopeRF RFM300W** в составе миниатюрной цифровой радиостанции.

Для записи и воспроизведения речи используются интегрированные в микроконтроллер аудиокодек или 12-битные АЦП и ЦАП. Трансивер CMT2300A функционирует в «прозрачном» режиме, модуляция и демодуляция осуществляются программными средствами микроконтроллера. Скорость передачи данных в радиоканале составляет 5 кбит/с.

Для сжатия речи применяется алгоритм AMR-4750 с математикой с плавающей точкой одинарной точности, аппаратно поддерживаемой ядром Cortex M4F. Качество речевого сигнала соответствует обычной GSM связи.

Такая радиостанция позволяет организовать симплексную радиосвязь на расстояниях до 300 метров и, при необходимости, обеспечить шифрование голоса с нужным уровнем защиты. Сверхнизкий ток потребления в режиме дежурного приема гарантирует длительную работу от одного комплекта батарей. Доступность и невысокая стоимость используемых компонентов определяют привлекательность такого изделия для массового производства.

Для получения дополнительной информации о проекте, а также оперативной и всеобъемлющей технической поддержки по используемым радиокомпонентам, обращайтесь к инженерам компании «Гамма-Украина», которая является официальным дистрибьютором компаний Nuvoton и HopeRF. Мы обеспечиваем прямые поставки продукции по дилерским ценам, техническую поддержку от производителя и помощь в выборе современных комплектующих, максимально соответствующих требованиям заказчиков.

Получить дополнительную консультацию, а также полную и оперативную техническую поддержку по всей продукции компании Nuvoton можно у официального дистрибьютора в Украине — ООО «Гамма Украина»:

тел.: (056) 745-46-65,
(066) 173-26-79, (096) 480-38-65,
(0562) 36-09-41, (0562) 36-07-92,
http://www.microchip.ua

Литература:

1. Об утверждении плана использования радиочастотного ресурса Украины. http://search.ligazakon.ua/l_doc2.nsf/link1/KP060815.html
2. CMT2300AW — new 4-wire SPI 213-960Mhz RF Transceiver IC. http://www.hoperf.com/ic/rf_transceiver/CMT2300A.html
3. Гаевский С. Новый интегрированный трансиве CMT2300AW — универсальное решение для передачи данных на короткие дистанции. CHIP NEWS Украина, № 3 (173), апрель, 2018.
4. Гаевский С. Радиоудлинитель COM-порта с низким энергопотреблением. CHIP NEWS Украина, № 4 (194), май, 2020.
5. HOPERF & CMOSTEK RFPDK tool for Wirless RF Developing Stage. CMOSTEK RFPDK_V1.50. http://www.hoperf.com/information/tool.html
6. M451LG6AE NuMicro™ M451 Base series 32-bit microcontroller. https://www.nuvoton.com/products/microcontrollers/arm-cortex-m4-mcus/m451-base-series/m451lg6ae/
7. NUC505DSA NuMicro® NUC505 series 32-bit microcontroller. https://www.nuvoton.com/products/microcontrollers/arm-cortex-m4-mcus/nuc505-series/nuc505dsa/
8. Гаевский С. Голосовой коммуникатор на базе микроконтроллера Nuvoton M451LG6. CHIP NEWS Украина, № 1 (171), февраль, 2018.
9. Захаров В. Микроконтроллеры серии NUC505 на базе ядра ARM® Cortex™-M4 компании Nuvoton Technology. CHIP NEWS Украина, № 6 (156), август, 2016.
10. TS 26.104 ANSI-C Code for the floating-point Adaptive Multi-Rate (AMR) Speech Codec. https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=1400
11. Low power voice digital radio on Nuvoton NUC505 Microcontroller and CMOSTEK CMT2300A transceiver. https://github.com/gegel/NUC505_radio
12. NuMicro™ Family NuTiny-SDK-NUC505 User Manual. http://www.microchip.ua/nuvoton/UM_NuTiny-SDK-NUC505_EN_Rev1.01.pdf
13. Low power voice digital radio on Nuvoton M451 Microcontroller and CMOSTEK CMT2300A transceiver. https://github.com/gegel/M451_radio
14. NuTiny-SDK-M453 User Manualfor NuMicro ™ M453 Series. https://www.nuvoton.com/export/resource-files/UM_NuTiny-SDK-M453_EN_Rev1.00.pdf CNУ