Разработка модуля беспроводной передачи телеметрических данных в диапазоне частот 2,4 ГГц

(часть 2)

Александр Алый (Москва)

Для успешной и долгой жизни устройства, построенного на базе микроконтроллера, очень важно всегда иметь возможность его перепрограммирования в процессе эксплуатации. Для телеметрического модуля, используемого в системах охраны, контроля доступа и т.д., дополнительно возникает необходимость в защите кода от считывания и дизассемблирования. В данной статье описывается способ криптозащищённого перепрограммирования чипов серии MC1321x.

Введение

Возможность перепрограммирования модуля по месту установки и эксплуатации экономит значительные силы разработчиков и интеграторов на этапах ввода системы в эксплуатацию и приработки. Разработка надёжной технологии перепрограммирования – это то, с чего следовало бы начать разработку модуля для передачи телеметрических данных.

Когда количество установленных модулей становится достаточно большим, появляются риски утечки программного обеспечения, предназначенного для выполнения перепрограммирования, и самих программных кодов модулей. Это вызывает известные угрозы для информационной безопасности систем, построенных на перепрограммируемых модулях, и сужает сферу их применения. Криптозащищённый способ перепрограммирования, представленный в этой статье, решает описанную выше проблему.

Для того чтобы осуществить защищённое перепрограммирование микроконтроллера, в него предварительно записывается код специальной программы загрузчика. Программа-загрузчик занимает в памяти незначительное место и всегда получает управление после выполнения микроконтроллером аппаратного или программного сброса. Получив управление, загрузчик проверяет заданные условия активизации и либо отдаёт управление программе пользователя, либо остаётся в режиме загрузки, ожидая от внешнего устройства команды, управляющие программированием, и данные для программирования (в дальнейшем – программирующий поток). Команды и данные для программирования поступают в загрузчик в виде потока пакетов различной длины с зашифрованным содержимым. В данной статье приведена программа для PC, генерирующая и передающая в загрузчик программирующий поток через COM-порт компьютера. Программирующий поток может быть сохранён как файл с возможностью передачи на другие компьютеры для последующего программирования.

Зашифрованный в потоке код можно безопасно передавать в третьи руки, не опасаясь его дизассемблирования, копирования и модификации. Инструменты программирования, в свою очередь, не содержат никакой информации, способствующей расшифровке, и потому их передача тоже не несёт никакой опасности. Стойкость шифрации основана на общем секретном ключе шифрования, который хранится в памяти микроконтроллера, а также используется программой, генерирующей программирующий поток на PC. От считывания из памяти микроконтроллера ключ защищается установкой аппаратного бита защиты, а на стороне PC ответственность за конфиденциальность ключа несёт пользователь программы – генератора потока.

Программа-загрузчик защищается от модификации и считывания аппаратными средствами микроконтроллера и может быть только полностью стёрта вместе с кодом прикладной программы через интерфейс BDM (Background Debug Module) микроконтроллера.

Средства обеспечения защиты кода в чипах серии MC1321x

В микроконтроллерах семейства MC1321x реализован простой механизм защиты программы от считывания. В зависимости от состояния двух бит, находящихся в области Flash-памяти по адресу 0xFFBF, режим защиты от считывания либо включается, либо нет. От записи/считывания защищается ОЗУ микроконтроллера и память программ.

Важно, что в микроконтроллере есть несколько способов считывания областей памяти: через отладочный интерфейс BDM, с помощью выполнения программы в незащищённой от записи/считывания области и с помощью выполнения программы в защищённой от записи/считывания области. Последние два способа возможны благодаря фон-Неймановской архитектуре микроконтроллера. Незащищёнными от записи/считывания остаются области управляющих регистров. Однако, попытки считывания/записи защищённых областей из программы, расположенной в незащищённой области, или через интерфейс BDM блокируются.

Принцип работы загрузчика

Желательно, чтобы наличие загрузчика минимально влияло на структуру программы пользователя и не вводило дополнительных ограничений. Для этого в микроконтроллерах MC1321x есть механизм перенаправления векторов прерываний и защи-

ты областей памяти от модификации пользовательской программой.

Часть Flash-памяти микроконтроллера в верхних адресах до адреса 0xFFFF может быть после программирования защищена от последующей модификации установкой конфигурационных бит в этой же области памяти. Из этого следует, что данная область памяти уже никаким образом не может быть изменена программой, выполняющейся в самом микроконтроллере. Это гарантирует сохранность загрузчика в этой области при любом поведении пользовательской программы. Однако при этом блокируется и запись в область векторов прерываний, находящаяся в самых верхних адресах. Эта область, как правило, должна модифицироваться при загрузке пользовательской программы, и чтобы разрешить противоречие, в микроконтроллере реализован механизм перенаправления векторов прерываний после защиты верхних блоков памяти. Когда возникает прерывание при включенном перенаправлении, вектор прерывания извлекается из области памяти, которая находится непосредственно на стыке с защищённой от модификации памятью и повторяет структуру исходной области векторов прерываний.

Таким образом, зная размер защищённой памяти, пользователь в настройках среды разработки должен всего лишь указать линкеру другую область расположения векторов прерываний. При этом после загрузки программа останется работоспособной. Особенность в том, что вектор, по которому передаётся управление после сброса, не перенаправляется и после загрузки пользовательской программы по-прежнему указывает на точку входа в загрузчик. Загрузчик по-прежнему получает управление при старте программы и может быть использован для многократного перепрограммирования.

Описываемый загрузчик имеет размер чуть больше 4 Кб, и поэтому ему выделяется ближайший доступный размер защищаемого блока в 8 Кб. Блок защищённой от модификации памяти в этом случае размещается по адресам 0xE000 – 0xFFFF. Соответственно, область векторов прерываний для пользовательской программы начинается с адреса 0xDFC0. Загрузчик сконфигурирован так, чтобы при передаче управления пользовательской программе извлекать двухбайтный адрес перехода с адреса 0xDFFE, где должен находиться вектор перехода по сбросу пользовательской программы.

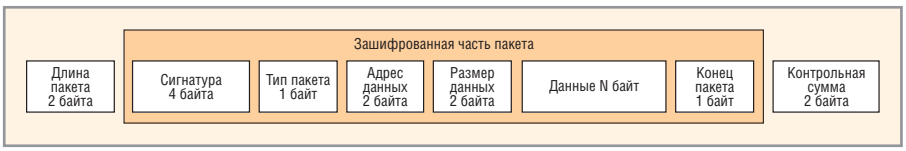

Рис. 1. Формат пакета

Загрузчик при старте проверяет контрольную сумму кода пользовательской программы и передаёт ей управление только в случае корректности контрольной суммы. Сама контрольная сумма формируется и записывается загрузчиком при программировании по адресу, на два байта меньше адреса начала области пользовательской программы.

Загрузчик также можно принудительно оставить работающим, если при выполнении сброса установить низкий логический уровень на линии 0 порта A.

Алгоритм шифрации

В качестве алгоритма шифрации использован алгоритм Rijndael, больше известный как AES – победитель конкурса, объявленного Американским национальным институтом стандартов, на лучший криптографический алгоритм 2000 г.

AES отличает достаточная простота и возможность распараллеливания операций при аппаратной реализации. AES применяется для защиты данных по спецификации ZigBee и во множестве других приложений. Реализация AES в загрузчике может освободить от необходимости его реализации в прикладной программе.

Алгоритм относится к классу блочных алгоритмов с симметричным ключом. Это означает, что информация шифруется блоками (в данном случае по 16 байт) и для шифрования и дешифрования используется один и тот же секретный ключ. Ключ – это 16, 24 или 32 байта данных, сгенерированных генератором случайных чисел (лучше, если генератор не детерминированный). Чем меньше длина ключа, тем быстрее выполняется алгоритм AES, но при этом уменьшается криптостойкость шифрованных данных. В приведённом проекте загрузчика применён 32-байтный ключ. Микроконтроллер MC1321x довольно быстро выполняет вычисления AES, и длина ключа сравнительно мало влияет на скорость загрузчика.

Кроме самого алгоритма шифрования важен режим работы блочного шифра. Поскольку поставлена цель не только не дать расшифровать код, но и не дать его подменить, то изменение хотя бы одного бита в потоке должно привести к сбою при дешифровании и искажению всей последующей информации. То есть злоумышленник не должен иметь возможности преднамеренно исказить информацию в одном определённом маленьком блоке. Это достигается введением взаимосвязи шифрования одних блоков от других. В загрузчике применён режим под названием «сцепление шифрованных блоков» (Cipher Block Chaining, CBC). В этом режиме каждый блок открытой информации складывается с помощью операции XOR с предыдущим блоком шифрованной информации.

Блочный шифр требует, чтобы длина информационных блоков была кратна размеру шифруемого блока.

Описание параметров в конфигурационном файле

Имя параметра	Описание
PAGE_SIZE	Размер страницы Flash-памяти микроконтроллера
MEM_SIZE	Размер доступной Flash-памяти микроконтроллера
CRC_ENABLE	Разрешение использования контрольной суммы при передаче пакетов
KEY1	Первая часть ключа шифрования
KEY2	Вторая часть ключа шифрования
KEY3	Третья часть ключа шифрования
INITIAL_VECTOR	Начальный код для инициализации алгоритма шифрования
SIGNATURE	Код подписи, сопровождающей каждый пакет

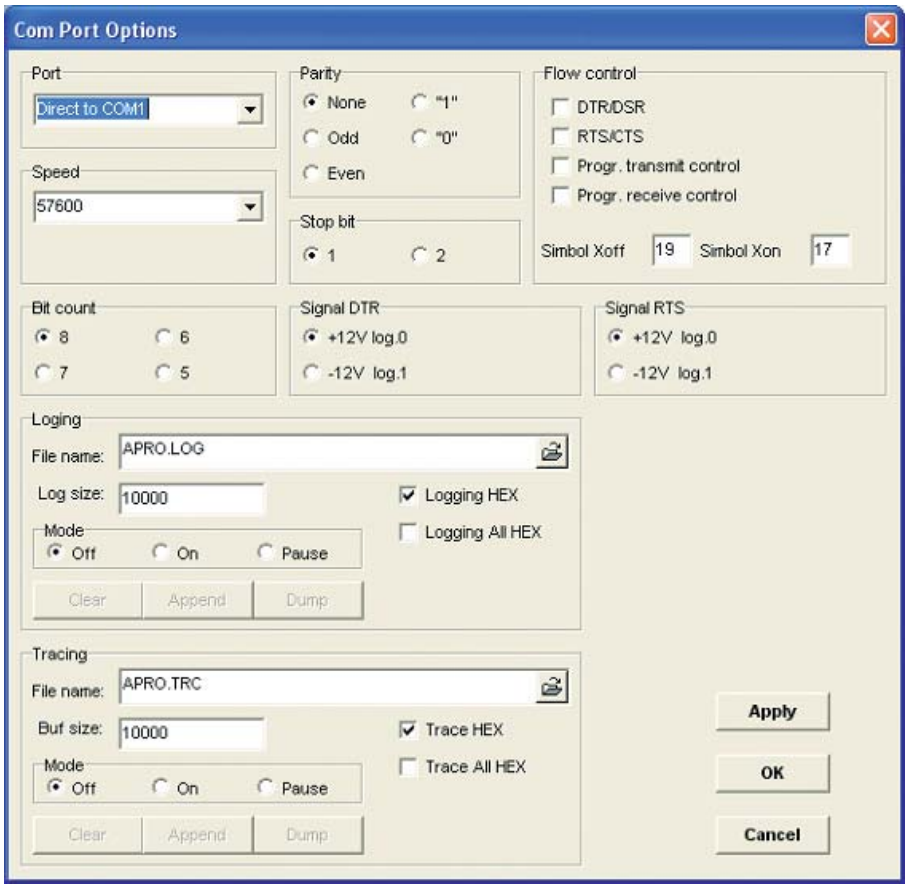

Рис. 2. Вид окна настроек COM-порта

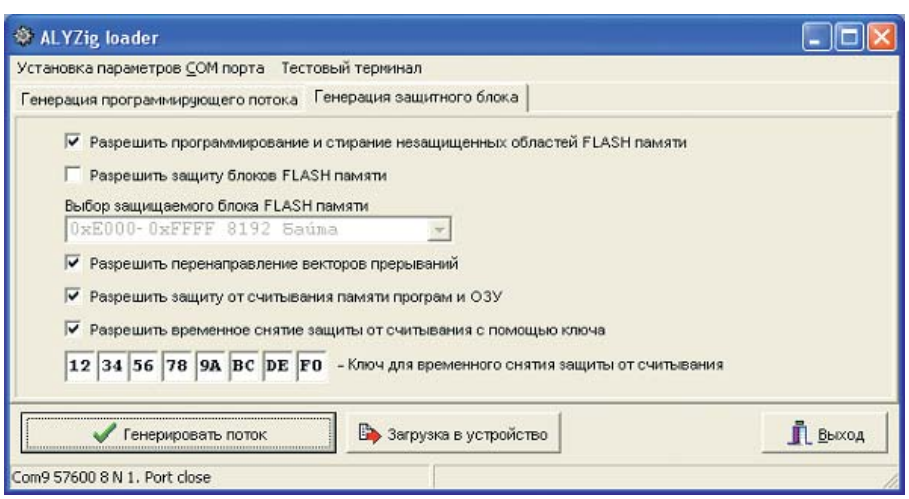

Рис. 3. Вид закладки для генерации защитного блока

Если это условие не соблюдается, то необходимо блоки информации дополнять до заданной длины. В загрузчике используется дополнение данными с выхода генератора псевдослучайных чисел.

После дешифрования пакетов с информацией загрузчик проверяет корректность специальной сигнатуры в начале пакета. Сигнатура – это четыре заранее определённых байта, которые вставляются в пакет при шифровании. Сохранность сигнатуры означает, что пакет был расшифрован правильно.

Реализация AES в микроконтроллере MC1321x занимает 1342 байта Flash-памяти и требует 776 байт ОЗУ.

Протокол обмена

Протокол обмена загрузчика очень прост. Пакеты потока последовательно высылаются из PC по последовательному интерфейсу в микроконтроллер. После каждого пакета ожидается один байт ответа от микроконтроллера. Ответ может либо быть положительным, либо представлять код ошибки. Ожидание ответа может длиться некоторое время, по истечении которого программой PC фиксируется ошибка.

Программирующий поток состоит из пакетов, которые имеют формат, представленный на рис. 1. Пакеты могут передавать команды или данные и могут быть следующих типов:

- передача блока программируемых данных,
- передача команды предварительной подготовки страницы программируемых данных,
- передача команды на программирование данных,
- передача команды на стирание страницы Flash-памяти,
- передача команды на запись контрольной суммы,
- передача команды на сброс микроконтроллера.

Программное обеспечение

В дополнительных материалах к статье на сайте журнала можно найти программу ALYZil.exe, которая работает как генератор программирующего потока и как программатор устройства. Программа разработана для выполнения в среде Windows XP. Вместе с программой также публикуются её исходные тексты для среды разработки Delphi 7. В качестве входных данных при генерации программирующего потока программа принимает файлы в стандарте Intel HEX или Motorola S-records. Программа снабжается конфигурационным файлом. Содержимое файла представляется в виде текстовых строк с парами: имя параметра – значение параметра. Описания параметров в конфигурационном файле приведены в таблице.

В режиме программирования программа может просто считывать файл потока и направлять его в устройство через COM-порт компьютера. Для этого поле «Файл загружаемой программы» должно быть пустым. Стоит отметить, что COM-порт может быть и виртуальным, например, реализованным на USB-интерфейсе, или располагаться на удалённом компьютере через механизм совместного использования портов. Трудностей при этом никаких не возникнет, поскольку в программе не используются модемные сигналы COM-порта и сделаны достаточно длительные тайм-ауты.

Для подключения к устройству необходимо настроить параметры

COM-порта. Для этого выбирается пункт меню «Установка параметров COM-порта». Неопытным пользователям следует оставить все настройки, как показано на рис. 2, кроме номера COM-порта, который необходимо правильно определить самому пользователю.

Для генерации программирующего потока необходимо перейти на закладку «Генерация программирующего потока» и в поле «Файл конфигурации» ввести путь и имя конфигурационного файла, а в поле «Файл загружаемой программы» ввести путь и имя файла с расширением HEX или S19, содержащего код программы. В поле «Файл программирующего потока» надо ввести имя выходного файла и нажать одну из двух кнопок в зависимости от желаемого действия. Кнопка «Генерировать поток» создаёт файл потока, но программирование в устройство не производится. Кнопка «Загрузка в устройство» генерирует поток и запускает его передачу в устройство. Если поле «Файл загружаемой программы» в этот момент пустое, то поток не генерируется, а производится попытка его считывания из файла, указанного в поле «Файл программирующего потока».

При генерации также нужно определить две опции. Если опция «Стирать страницы перед программированием» отмечена, то перед программированием 512-байтных страниц Flash-памяти микроконтроллера загрузчик их будет полностью стирать. Если опция не отмечена, то загрузчик сохранит содержание страницы по тем адресам, на которые не накладывается новая информация при программировании страницы. Отмеченная опция «Не шифровать выходной поток» отключает шифрацию, позволяя проверить корректность формирования выходного потока.

При использовании во время разработки недорогого внутрисхемного отладчика Multilink у пользователя нет удобной возможности установить необходимые атрибуты защиты и конфигурации памяти. Поэтому после записи самого загрузчика в память микроконтроллера он остаётся незащищённым от случайных модификаций.

Для правильной конфигурации областей, занятых загрузчиком, и их защиты от модификации программа в закладке «Генерация защитного блока» позволяет установить необходимые биты и запрограммировать соответствующую страницу во Flash-памяти микроконтроллера. Эти действия следует проделать прежде, чем программировать прикладной код в микроконтроллер.

На рис. 3 показаны опции генератора защитного блока. Загрузчик располагается в верхних областях памяти микроконтроллера, начиная с адреса 0xE000. Поэтому необходимо включить опцию «Разрешить защиту блоков Flash-памяти» и выбрать блок размером 8192 байта. Также необходимо включить опцию «Разрешить перенаправление векторов прерываний». По желанию пользователя опция «Разрешить защиту от считывания памяти программ и ОЗУ» может оставаться не включенной. Ключ для временного снятия защиты от считывания может содержать только шестнадцатеричные цифры и должен быть сохранён пользователем отдельно, поскольку программа его нигде не запоминает.

Важно помнить, что после защиты блока Flash-памяти изменить заданную конфигурацию из данной программы будет невозможно. Единственной возможностью для этого остаётся полное стирание всей Flash-памяти микроконтроллера с помощью BDM-отладчика и перепрограммирование (загрузчика в том числе).

В дополнительных материалах на сайте журнала также находятся исходные тексты загрузчика. Они представлены в составе файлов проекта для среды разработки CodeWarrior IDE версии 5.7.0 build 2015 с компилятором и линкером версии 5.0.14.6124 для семейства HC08. Загрузчик разработан для платы 13213-NCB (Network Coordinator Board) из состава оценочного набора фирмы Freescale, но особенности аппаратной платформы очень мало отражаются на исходных текстах, и поэтому их можно считать достаточно универсальными для семейства микроконтроллеров MC1321x.

Для конфигурации области расположения загрузчика в памяти микроконтроллера используется файл Project.prm. Объём памяти, занимаемый кодом загрузчика, адреса процедур, компоновку и глубину использования стека можно просмотреть в файле Project.map.