1) Добрый день меня зовут Чеботарев Георгий Михайлович, студент группы 43501/4. Мой научный руководитель Богач Наталья Владимировна и тема моей работы: разработка помехоустойчивого кодека с применением технологий дизассемблирования и реверс-инжиниринга.

2) Основной проблемой передачи данных является наличие шума в канале. Как следствие на приемной стороне принимается уже не та посылка, что отправлялась. Одним из основных способов повышения надежности является кодирования. Выполняя кодирование данных перед оправкой, удается минимизировать воздействие шума на исходные данные. При декодировании возникшие ошибки обнаруживаются и частично или полностью исправляются.

3) Исходными данными проекта явился набор прошивок микропроцессора TMS320C54, широко используемого при обработке цифровых сигналов. Известно, что одним из компонентов прошивки является кодек, с двумя режимами кодирования. Однако двух режимов кодирования стало недостаточно для эффективной работы группы однотипных устройств на базе TMS320C54. Возникла необходимость модернизировать кодек, внедрив промежуточный режим кодирования. Данный режим придаст гибкость всей системе кодирования. Модернизированный кодек должен быть совместим с исходным. Это позволит модернизировать устройства постепенно, не разделяя все устройства на две отдельные группы.

   Для выполнения поставленной задачи, разработка была разделена на несколько этапов

4) Первый этапом стало дизассемблирование. Дизассемблирование – это трансляции бинарного файла на язык программирования ассемблер. Анализируя ассемблерный код, можно делать выводы о внутренних процессах бинарного файла. В качестве дизассемблера была использована среда IDA PRO, а Code Composer Studio был выбран в качестве симулятора. Выбор обусловлен, во-первых, качеством данных сред, а во-вторых наличием поддержки искомого микропроцессора.

5) Комбинируя работу дизассемблера и симулятора, загружая прошивки с различными внутренними состояниями, и анализируя ход выполнения программы, удалось с точностью до команды определить функции, принадлежащие кодеку. Проводить проверку всей прошивки является затратным процессом, поэтому этап тестирования настал только после обнаружения кодека. Тестирование подтвердило работоспособность и полноту функций кодека. Он состоит из 12 основных функций, которые условно можно разделить на 4 группы: сверточный кодер, декодер-Витерби, и два модуля чередования.

Сверточный кодер кодирует данные в двух режимах: двумя или четырьмя битами. В основе кодера находится конечный автомат на 64 состояния;

Декодер – декодирует данные и исправляет некоторый % ошибок, возникших при передаче. В основе декодера находится алгоритм Витерби с мягкими весами. Глубина декодирования в данной реализации составляет 6, то есть при поиске правильного варианта декодирования, отслеживается 64 лучших пути декодирования, которые постоянно обновляются. Остальные отбрасываются. После нахождения лучшего вариант декодирования, выполняется восстановление исходной последовательности.

Для того, чтобы пояснить смысл модулей чередования, необходимо вспомнить процессы, происходящие в канале.

При передаче, в следствии воздействии шума, происходит частичная потеря данных. При этом ошибки в канале возникают подряд, целыми последовательностями. Декодирующие способности алгоритма Витерби на прямую зависят от длины таких ошибочных последовательностей. Чем больше длина пачки ошибок, тем меньше шансов, что Витерби успешно восстановит исходную последовательность.

Модуль чередования – смешивает несколько закодированных сообщений. Поток образовавшихся данных отправляется в канал. На приемной стороне, модуль ликвидации чередования возвращает правильный порядок бит, при этом достигается дробления пачек ошибок на более мелкие объединения (1,2 бита);

6) После идентификации внутренней структуры, была применена технология реверс инжиниринга. Кодек был переписан с языка программирования ассемблер на язык программирования Си. Функциональность при этом изменена не была. Кодек был оформлен в виде библиотеки, включающей 4 идентичных-основополагающих модуля: кодер, декодер и два модуля чередования.

7) После написания было проведено тестирование согласно приведенному алгоритму. Генерируется случайная последовательность, затем она обрабатывается кодером, потом декодером, после сравнивается с исходной. Если последовательности не совпали – тест помечается как неудавшийся, и проводится ручная отладка программы. Благодаря тестированию удалось выявить три ошибки на разных этапах и режимах работы кодека.

8) Модернизация кодека заключалась во внедрение промежуточного режима: кодирование тремя битами. Организация кодирования тремя битами в ряде случаев обещала повысить эффективность работы кодека, позволив сократить временные затраты на кодирование и передачу.

Для внедрения нового режима модули чередования изменять не пришлось. В модули кодирования и декодирования были внесены небольшие

дополнения. Серьезной проблемой на данном этапе стало описание переходов конечного автомата, при использовании трех полиномов. Сложность заключалась в понимании реализации построения решетки Витерби в декодере. Но через некоторое время проблема была успешно решена. По завершению разработки было вновь проведено тестирования. Оно подтвердило работоспособность как старых режимов, так и нового.

9) Последним этапом работы стала проверка граничных значений восстанавливающих способностей. Для этого был модернизирован используемый ранее алгоритм тестирования. Как и прежде генерируется 1024 случайных последовательностей по 240 бит. Последовательности кодируются. Затем в каждую из закодированных посылок вносится одинаковое количество ошибок (ось X). После сообщение декодируется и подсчитывается процент удачных тестов (ось Y). Затем количество ошибок инкрементируется и тест повторяется вновь, до отказа работы декодера. Из графика видно следующие:

при передаче 1 посылки:

режим 2 способен исправить до 200 ошибок, что соответствует 13% длины передаваемого сообщения,
режим 3 – до 360 ошибок – это 21%,
режим 4 – до 400 ошибок и это 20% от длины передаваемого сообщения.

Очевидно, что режим 3 крайне эффективен. В 80% случаев, где раньше требовалось использовать режим 4, теперь можно будет использовать режим 3. Размер передаваемого сообщения при этом сократился в 0,75 раз, что позволит сократить временные и ресурсные затраты, без потери качества.

10) К результатам можно отнести следующие пункты. Основная задача была выполнена: прошивка была дизассемблирован, кодек обнаружен и модернизирован, после чего оформлен в виде библиотеки.

Спасибо за внимание.