1) Добрый день меня зовут Чеботарев Георгий Михайлович, студент группы 43501/4. Мой научный руководитель Богач Наталья Владимировна и тема моей научной 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. Один из основных способов повышения надежности – это 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. Одним из компонентов прошивки является кодек. Известно, что кодек имеет два режима кодирования. Однако двух режимов стало недостаточно для эффективной работы группы однотипных устройств. Возникла необходимость создать кодек, который включал бы в себя как два основных режима, так и промежуточный, который придавал бы гибкость всей системе кодирования. Разработанный кодек должен быть совместим с исходным кодеком, при работе двух и трех режимов. Это позволит модернизировать устройства постепенно, не разрывая всю сеть устройств на два отдельных лагеря.

   Для выполнения поставленной задачи, разработка была разделена на несколько этапов

4) Первый этапом является дизассемблирование. Дизассемблирование – это трансляции некоторого бинарного файла на язык ассемблер. В качестве дизассемблера была использована среда IDA PRO, а Code Composer Studio был выбран в качестве симулятора. Выбор обусловлен, во-первых, качеством и надежностью данных сред, а во-вторых наличием поддержки искомого микропроцессора.

5) Комбинируя работу дизассемблера и симулятора, загружая прошивки с различными внутренними состояниями, удалось с точностью до команды определить функции, принадлежащие кодеку. После этого было проведено тестирование обнаруженного кодека. Тестирование подтвердило работоспособность и полноту функций кодека. Кодек состоит из 12 функций, которые условно можно разделить на 4 группы: сверточный кодер, декодер-Витерби, и два модуля чередования.

   Кодер – кодирует данные двумя или четырьмя битами;

Декодер – декодирует данные и исправляет некоторый % ошибок. В основании декодера лежит алгоритм Витерби с мягкими весами. Т.е. каждый принятый бит характеризуется некоторым весом (вероятностная мера, что это 0 или 1);

Модуль чередования – смешивает несколько закодированных сообщений. Поток образовавшихся данных отправляется в канал, где в следствии шума происходит частичная потеря данных. На приемной стороне модуль ликвидации чередования – возвращает правильный порядок бит. Таким приемом удается добиться дробления ошибок, возникших в канале. Дело в том, что в канале биты чаще всего портятся подряд, что резко ухудшает декодирующие свойства алгоритма Витерби.

Сверточный кодер работает в двух режимах, выполняя кодирование двумя или четырьмя битами.

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 должно повысить эффективность работы кодека, позволив сократить временные затраты.

9) Для проверки граничных значений восстанавливающих способностей 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). 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 эффективен. В 80% случаев, где раньше потребовалось бы использовать режим 4, теперь можно использовать режим 3. Размер передаваемого сообщения сократится в 0,75 раз. Это позволит сократить временные и ресурсные затраты, без потери качества.