Описание программы приема сигналов Op32Rx v1.7.0.

Автор: RN3AUS/Alex

1. Общие положения.

Внимание:
- программа не претендует быть заменой "стандартной" Opera v1.5.x by EA5HVK;
- программа не преследует целью превзойти качество обнаружения слабых сигналов OPDS by DF6NM.

Автор выражает признательность Jose EA5HVK и G0NBD за разработку и реализацию превосходного режима Opera.

Автор выражает признательность Markus DF6NM за разработку и предоставление открытых исходных кодов превосходной программы OPDS. Многие идеи и программные решения заимствованы из opds2h5c.bas

Автору неизвестно, как устроена программа Opera v1.5.x by EA5HVK. Все возможные алгоритмические совпадения непреднамеренны и вполне естественны.

Автором сделана попытка реализовать мягкое списочное декодирование максимального правдоподобия, снабдив программу дополнительными удобными опциями. Автор надеется, что программа будет полезна в тех случаях, когда по каким-либо причинам затруднительно использовать Opera v1.5.x by EA5HVK или OPDS by DF6NM.
Программа предлагается по принципу «как есть» без каких-либо гарантий. Исходные коды декодера открыты и могут использоваться всеми желающими.

2. Основные сведения о программе.

Программа предлагается в двух вариантах (v1.7.0 – только GUI):
- консольном Op32RxC.exe (удобно использовать на грабберах)
- GUI Op32Rx.exe (для наблюдений и экспериментов)

Обработка сигнала в обеих версиях происходит одинаково.
В графической (GUI) версии дополнительно поддерживается сохранение скриншотов.

3. Что делает программа.

Программа обрабатывает отсчеты сигнала со звуковой карты или из wav-файла, демодулирует их и пытается декодировать сигнал в формате Opera-32. Если декодер обнаружил совпадение контрольной суммы, декод записывается в текстовый файл, а также отправляется рапорт на pskreporter. Программа имеет встроенный автоматический подавитель импульсных помех (NB).

4. Чем отличается от "стандартной" Opera v1.5.x by EA5HVK:

- имеет открытый исходный код на C++ (CBuilder);
- работает с произвольной Fc = 500…23500 Гц;
- кроме отправки спотов на сайт pskreporter.info, сохраняет декоды в текстовом файле (как detected.txt в opds)
- использует список частот для глубокого поиска сигнала,
  этот список автоматически пополняется при каждом успешном декодировании
- можно задать список игнорируемых частот (например, линий Loran)
- правильность декодов может проверяться по списку известных корреспондентов (как callsloc.txt в opds)
- может декодировать сигнал из wav-файла
- реализован вариант работы декодера со списочным декодированием
  (с коротким и длинным списком, соответственно глубокий и сверхглубокий поиск).
  Это позволяет правильно декодировать очень слабые или замирающие сигналы, что не всегда удается сделать стандартным декодером. (Впрочем, автору неизвестно, как реализован декодер в оригинальной программе Opera v1.5.x by EA5HVK; возможно, там это сделано лучше.)
- отображает спектр с длительным накоплением

- позволяет автоматически сохранять скриншот водопада спектра в момент декодирования с отметкой на скриншоте трассы сигнала.
- расходует значительно меньше памяти (7-12 MB) вместо 75-250 MB
- не использует никакой дополнительной информации, поступающей из Интернета
- имеет множество настроек, что позволяет оптимизировать использование ресурсов компьютера.
- возможно, способна обнаружить более слабые сигналы.

5. Чем отличается от OPDS by DF6NM:
- работает автономно, не нуждается в сторонних программах обработки сигнала со звуковой карты (SpectrumLab)
- можно в режиме offline обрабатывать предварительно записанные wav-файлы
- осуществляет настоящее декодирование, а не корреляционный выбор наиболее похожей копии известного сигнала
- может декодировать любой не известный заранее позывной
- отправляет споты в Интернет
- можно назначить список частот для глубокого поиска сигнала,
  этот список автоматически пополняется при каждом успешном декодировании.
- в случае двухкратного декодирования позывного, отсутствующего в листе callsloc.txt, автоматически запоминает его в файле known_list.txt
- отслеживает медленный дрейф и блуждания частоты передатчика/приемника без ухудшения качества декодирования при расстройках до +-0,85 Гц.
- отслеживает различие тактовых частот передающей и приемной сторон.

6. Как работает программа.

Сигнал берется со звуковой карты. Если таких устройств несколько, есть возможность выбрать нужное.

Частота дискретизации 12 кГц, 16 бит (или 48 кГц, 16 бит для Fc > 5500 Гц). В настройках можно выбрать центральную частоту НЧ сигнала, соответствующую центру полосы 137500 Гц. Например, при Dial=136000 Гц центральная частота спектра OP-32 сегмента будет равна Fc=1500 Гц. Есть возможность установить Fc от 500 Гц до 23500 Гц. Это делается в ini-файле. (Далее мы рассмотрим все настройки в этом файле подробно.)

Определяется средний уровень сигнала. Если присутствуют импульсы, превышающие утроенное значение среднего уровня, импульс подавляется (NB).
Далее очищенный от импульсных помех сигнал фильтруется цифровым КИХ-фильтром с полосой 200 Гц по уровню -3 дБ.

Для трех номиналов "стандартных" Dial (135000 Гц, 135500 Гц и 136000 Гц, с возможностью точной подстройки в пределах плюс-минус 10 Гц) применяются заранее рассчитанные фильтры Баттерворта 6 порядка с 255 отводами, имеющие хорошую равномерность в полосе пропускание и большое подавление за ее пределами.
Если же Dial нестандартный, то КИХ фильтр рассчитывается средствами программы (методом оконного взвешивания с окном Блэкмана-Харриса). Такой фильтр имеет заметно большую неравномерность в полосе пропускания. Количество "отводов" определяется автоматически и не превышает 1024 (в реальности получается около 270).

Отфильтрованный сигнал гетеродинируется и переносится по частоте вниз на центральную частоту 150 Гц (весь спектр лежит в пределах от 100 до 200 Гц).
Для подавления побочных продуктов преобразовния служит еще один КИХ-фильтр с 64 отводами.

Далее осуществляется первая децимация в 24 раза, теперь отсчеты следуют с частотой 500 раз в секунду.

Для осуществления узкополосной согласованной фильтрации в программе применяется линейка из цифровых рекурсивных БИХ фильтров 2-го порядка (цифровой колебательный контур) с полосой пропускания 0.122 Гц по уровню -3дБ (что равно 1/8.192 с - обратная величина длительности одной посылки). Количество фильтров в линейке 1230, что несколько избыточно: в принципе, достаточно иметь 100 / 0.122

= 820 фильтров для перекрытия всего диапазона 100 Гц. Однако эксперименты показали, что чувствительность приема возрастает, если фильтры "стоят" несколько плотнее; в результате была выбрано полуторное количество фильтров.

Может показаться странным применение "цифровых колебательных контуров" там, где обычно применяют быстрое преобразование Фурье. Возможно, вычислительная эффективность была бы несколько выше, но так как на фильтры поступают отсчеты после децимации, то выигрыш в производительности был бы не велик. С другой стороны, наглядность при использовании фильтров получается выше, можно экспериментировать с детектированием и количеством фильтров, чего мы были бы лишены при FFT. Кроме того, FFT уже «набило оскомину»; гораздо интереснее пойти нестандартным путем.

После узкополосных фильтров стоят детекторы. Это может быть либо простой квадратичный амплитудный детектор (когда входной отсчет фильтра перемножается сам на себя), либо квазикогеррентный квадратурный детектор. Эксперименты показали, что оба детектора приблизительно эквивалентны в плане чувствительности, однако когеррентный детектор расходует больше вычислительных ресурсов. По умолчанию когерентный детектор отключен. Затем сигнал фильтруется ФНЧ (БИХ-фильтр эквивалентный сложной RC-цепочке) с частотой среза 0,25 Гц.

Далее происходит децимация сигналов с выходов детекторов еще в 1024 раза, так что на длительность одной посылки приходится ровно 4 отсчета (около 1 отсчета за 2 секунды).

Расфильтрованные и продетектированные сигналы поступают в буфер памяти, имеющий 1230 каналов и длиной 240 (количество посылок Opera плюс одна) * 4 = 960 ячеек. Именно этот буфер и занимает основное место в оперативной памяти 1230 * 960 * 4 (размер float в байтах) = 5 МБайт.

Далее обрабатываются только отсчеты из этого буфера, хранящего в себе информацию за последние 32 минуты.

По каждому каналу (под термином канал понимается выход согласованного фильтра) вычисляется текущая сумма, что соответствует энергии сигнала на длительности передачи Op-32 (32 минуты). Эти значения в дальнейшем используются для отображения амплитудного спектра и водопада.

Для уменьшения количества вычислений декодированию подвергаются не все каналы, а лишь те из них, где есть максимум накопленного сигнала. Максимумы ищутся следующим образом. Каждое значение сравнивается с несколькими соседними значениями (количество их задается параметром MIN_PEAK_SPACE). Если соседние значения меньше текущего – то это максимум. Таких максимумов в спектре очень много. Чтобы сократить их количество, применяется следующий алгоритм. Пусть задано желаемое количество максимумов для обработки (параметр N_MAX). Если найдено больше максимумов, чем требуется, то выполняется «прореживание»: весь спектр разбивается на несколько участков, в каждом из них удаляется максимум, имеющий наименьшую амплитуду. Так повторяется до тех пор, пока не останется требуемое количество максимумов. Разбивка спектра на участки выполняется из следующих соображений. Если входной фильтр имеет большую неравномерность, или присутствует сильный сигнал, то в часть максимумов, лежащая ближе к «горбу» фильтра будет всегда иметь большую амплитуду, чем все остальные (то есть сама «шумовая полка» будет иметь «горб»). Если просто выбирать определенное количество максимумов с наибольшей амплитудой, все они будут сгруппированы либо около мощного сигнала, либо около «горба» фильтра. Это нехорошо, так как наиболее интересные слабые сигналы от далеких корреспондентов останутся без внимания. Применение описанного выше алгоритма «прореживания» максимумов позволяет вести отбор, когда все они находятся в примерно равных условиях.

Обычно достаточно установить параметр N_MAX=30…40. Если же вычислительные ресурсы не очень критичны, то проще указать N_MAX=-1, и программа будет обрабатывать все найденные (даже маленькие) максимумы.

Из числа каналов, выделенных для дальнейшей обработки, исключаются те, частоты которых совпадают с перечнем частот из файла loran_list.txt (если

параметр IGNORE_LORAN_LINES=1).

Если указано USE_FREQUENCY_LIST=1, то на обработку берутся каналы, соответствующие частотам из файла freq_list.txt, независимо, есть ли на них спектральные максимумы или нет. Кроме того, для этих каналов включается режим списочного декодирования с длинным списком (Ultra Deep Decoder).

Из оставшихся каналов выбираются те, уровень сигнала в которых превышает уровень в соседних «невыбранных» каналах на установленную в настройках величину (параметр PEAK_THRESHOLD, указывается в дБ).

Декодер реализует алгоритм максимального правдоподобия, осуществляя перебор всех вариантов кодовых комбинаций, используемых Opera. Таких комбинаций 17, каждая комбинация имеет 8 вариантов. (Более подробно о структуре и способе формирования сигнала Opera можно прочитать в файле opera_protocol.pdf). Вычисляются корреляционные суммы, из них выбираются наибольшие, а также вторые после них. Вычисляется достоверность каждой комбинации. Формируется принятый информационный вектор, имеющий наибольшую корреляционную сумму, а также список его вариантов, с заменой части бит на менее вероятные (из вторых по корреляции кодовых комбинаций). Первый список состоит из 83 вариантов (1 самый вероятный + 17 вариантов с заменой 1 кодовой комбинации на менее вероятную + 1 вариант с заменой всех кодовых комбинаций + 64 комбинации с заменой двух наименее надежных комбинаций на все возможные варианты). Второй список включает в себя первый, а также дополнительно составляется перечень из замен двух кодовых комбинаций одновременно, замен трех наименее надежных комбинаций на все возможные; всего 602 варианта. Можно было бы строить и еще более длинные списки, но при этом начинается резко увеличиваться количество ложных декодирований.

Отметим, что декодер учитывает возможность некоторого расхождения тактовых частот приемной и передающей сторон, так что суммарная длительность сигнала может быть несколько длиннее или короче ожидаемой. Осуществляется поиск в пределах плюс минус длительности одной посылки (8.192 с), что соответствует расхождению тактовых частот 1/239=0,42%= 4184 ppm (это довольно много).

Кроме того, декодер имеет режим слежения за медленным дрейфом частоты сигнала, для чего отслеживается постепенное смещение спектрального максимума в пределах до плюс-минус 0,854 Гц от его начального положения. Точнее, ширина полосы поиска равна удвоенной величине (MIN_PEAK_SPACE * 0,122) Гц. По этой причине не следует задавать минимальное расстояние между пиками задавать меньше 3. Рекомендуемое значение 4 или 5.

Декодер осуществляет перебор вариантов до тех пор, пока список не закончится, либо не произойдет совпадение контрольной суммы декодированной информации (позывной, передаваемой в Opera, защищается контрольной суммой CRC-16 (16 проверочных бит) и второй контрольной суммой CRC-16, из которой передается 3 бита). Если совпали обе контрольные суммы, то декодер возвращает декодированный позывной. Если совпала только одна (внутренняя 16 бит) контрольная сумма, то декодер помечает этот позывной как недостоверный (со знаком вопроса). Этот декод принимается во внимание, если при этом соотношение сигнал-шум на выходе фильтра превысило некоторый порог (OP_THRESHOLD=3 – порог равен 3 дБ).

Что представляет собой это отношение сигнал-шум. Для наиболее совпадающего варианта сигнала имеется корреляционная сумма – это энергия нашего сигнала. Одновременно мы имеем величину энергии сигнале в этом частотном канале – это есть энергия сигнала + удвоенная энергия шума (шум есть и во время посылок и в паузах между ними). Отсюда находим отношение сигнал шум Eb/No=10*log(2.0*Psn/(Psn-Ps)), где Psn – мощность смеси сигнала и шума, Ps – мощность сигнала (корреляционная сумма декодера). Численные значения Eb/No по результатам эксперимента лежат в диапазоне от +4.0 дБ и выше, при этом сигнал, удовлетворительно видный на спектре и дающий декод в «стандартной» Opera -36 dB (opds -37.1db0p) имеет Eb/No +4.8 dB. Слабые сигналы, которые можно декодировать уже на самой грани возможностей глубокого декодера и которые уже не всегда декодируются Opera, имеют Eb/No около +4 dB. Иногда случались правильные декоды и с Eb/No чуть ниже +3.5 dB, однако при этом идет множество ложных декодов (но однако с правильными контрольными суммами!). Поэтому порог для подозрительных

декодов разумно установить примерно 4 dB).

Отдельно нужно оговориться, что, возможно, не совсем правомерно использовать здесь термин «Eb/No» - нормированное битовое отношение сигнал-шум. Однако, кажется, по смыслу измеренная нами величина близка к смыслу термина «Eb/No»; кроме того, это обозначение достаточно наглядно, поэтому я решил воспользоваться им и в дальнейшем.

Итак, на выходе декодера мы имеем декодированный позывной, с правильными контрольными суммами (суммой). Однако это еще не значит, что он действительно присутствовал в эфире. Практика показывает, что ложное обнаружение по шумам произвольных позывных с совпадающими CRC происходит довольно часто, порядка одного раза в полчаса/час. Можно было бы защититься от ложных декодов, введя сравнение с некоторым порогом и не регистрировать декод, если Eb/No ниже этого порога. Однако не так редки случаи, когда Eb/No правильного декода на слабом сигнале бывал ниже Eb/No ложного декода! Нас же интересует прием именно наиболее слабых сигналов, не хотелось бы потерять их, установив порог таким образом, чтобы надежно не пропускать ложные декоды. Возникает классическая дилемма: либо пропуск сигнала, либо ложная тревога. Кроме того, на практике часто получается следующая ситуация: если в эфире присутствует чей-то сильный сигнал, то на нем при ультра глубоком поиске легко «набираются» ложные декоды с очень хорошим уровнем Eb/No, так что сравнение с порогом здесь никак не поможет.

В программе реализован иной метод защиты от ложных тревог. Нужно согласиться, что в ДВ эфире работает очень немного корреспондентов, их позывные известны, появление каждого нового корреспондента не проходит незамеченным. Для программы OPDS by DF6NM используется файл callsloc.txt, где указаны позывные и локаторы всех реально работающих или когда-то работавших корреспондентов. Поэтому для «отсеивания» ложных декодов (а это всегда очень своеобразные позывные!) используется поиск декодированного позывного в файле callsloc.txt. Если он там есть, значит декод с большой долей вероятности правильный. Заодно и определяем расстояние до корреспондента.

Однако, вдруг в эфире присутствует новый корреспондент, которого пока еще нет в callsloc.txt? Было бы обидно не принять его только лишь по причине его отсутствия в файле. Для этих случаев предусмотрен механизм, когда дважды принятый на одной и той же частоте (± 2Гц) произвольный позывной считается отныне «известным» и он помещается в файл known_list.txt. Только локатор остается нам не известен. Мы можем, обнаружив декод от такого корреспондента, поместить его отныне в callsloc.txt, дополнив qth-локатором, либо, если это ошибка, стереть его из файла.

Мы можем использовать список частот freq_list.txt, так как многие из нас передают на одной и той же частоте. Кроме того, если частоты, на которой произошел правильный декод, нет в списке, она будет туда автоматически занесена.

Возможности декодера, судя по результатам экспериментов, ограничены величиной SNR порядка -39…-40 дБ в полосе 1 кГц. Для поиска более слабых сигналов в программе реализован корреляционный прием в двух вариантах. Это знаковый коррелятор и «мягкий» коррелятор, подобный OPDS.

Знаковый коррелятор относится к классу так называемых непараметрических или «робастных» алгоритмов, устойчивых к различным отклонениям законов распределения шума от нормального. Для осуществления знаковой корреляции сигнал с выхода детектора каждого канала сравнивается с плавающим порогом, устанавливаемым динамически таким образом, чтобы в результате образовывался поток нулей и единиц с примерно равным (в среднем) их количеством. Бинарные отсчеты сравниваются со всеми вариантами PIC-кодов, соответствующих позывным из файла callsloc.txt. Подсчитывается количество совпавших битов. Если это количество превысило порог коррелятора (CORRELATOR_237_THRESHOLD=161), то сигнал считается обнаруженным. Порог выбирается по необходимой вероятности ложной тревоги. Чем выше порог, тем менее вероятно, что из шумов наберется требуемое сочетание нулей и единиц. Рассчитать вероятность можно в окне дополнительных

настроек (Settings->Setup another settings…). На практике порог должен лежать в пределах от 159 (при этом ложное обнаружение будет происходить достаточно часто, по нескольку раз в сутки; но зато и чувствительность коррелятора будет высокой) и до 163 (ложное обнаружение по шуму будет крайне редким; чувствительность будет снижена на 1-2 дБ). В среднем рекомендуется устанавливать 161. Еще раз для ясности: это означает, что в принятом сигнале должны совпасть 161 позиция из 237-ми. Откуда это число 237? Ведь длина PIC-кода 239? Да, но первые две позиции всегда равны 1 для всех кодов, поэтому они исключаются из сравнения, остается 237.

Однако с обнаружением превышения порога обработка не заканчивается. Все было бы хорошо, если бы нам было нужно выделять сигнал лишь на фоне белого шума, зная момент начала передачи. Но это не так. Как выяснилось, очень часто порог будет превышен и в том случае, если в эфире присутствует сигнал (и он еще не закончился к данному моменту) одного из корреспондентов, а мы сравниваем его с PIC-кодом совершенно другого корреспондента. И вдруг многие биты совпадают и мы совершаем ложную фиксацию сигнала. В чем же дело? В больших так называемых боковых лепестках взаимокорреляционной функции этих PIC-кодов. Выяснено, что нередко между сдвинутыми на несколько позиций относительно друг друга различными PIC-кодами присутствует совпадение до 184 битов, что конечно намного выше нашего порога в 161, установленного для обеспечения заданной вероятности ложной тревоги по белому шуму. Что же делать?

В программе при фиксации факта превышения порога осуществляется проверка, что при сдвигах относительно текущего момента вперед и назад на +-230 позиций не находится лучшего совпадения с другими PIC-кодами. Если присутствует такой второй максимум и при этом совпадает не меньше битов, то декод считается ложным, вызванным взаимной корреляцией с еще не полностью принятым сигналом (или с уже прошедшим ранее). Остается лишь немного подождать…

Несколько иначе работает «мягкий» коррелятор (OPDS). В нем для каждого варианта PIC-кода вычисляется корреляционная сумма, когда отсчеты сигнала умножаются на элементы PIC-кода (1 понимается как +, а 0 как -) и суммируются. Соответственно, если 1 приходится на большой отсчет, а 0 на маленький, то сумма будет больше, чем в противном случае. Корреляционные суммы сравниваются между собой и из них выбирается максимальная и вторая по величине. Затем эти две суммы сравниваются. Если первая сумма больше второй в (OPDS_MAX1MAX2_THRESHOLD=1,6) раз, то «победивший» вариант PIC-кода передается на дальнейшую обработку. Опять, как и для битового коррелятора, выполняется поиск возможных совпадений сдвинутых копий сигнала с другими PIC-кодами. Если же наш максимум был «настоящий» и других лучших сдвинутых совпадений не найдено, то соответствующий позывной считается принятым.

Результаты работы корреляторов (а именно вычисленное битовое отношение сигнал-шум Eb/No), если обнаружено совпадение, сравнивается с порогом (OPDS_THRESHOLD=1,5 – в дБ). Если Eb/No сигнала больше порога, то фиксируется декод.

Декод выводится на экран и в лог не сразу. Примерно 10-12 секунд – несколько больше длительности одной посылки) – ожидается поступление новых декодов с данным позывным с лучшими результатами в плане отношения сигнал-шум. Дело в том, что успешное декодирование может происходить и при некоторых, в пределах длительности одной-двух посылок, сдвигах момента декодирования от истинного положения сигнала. Поэтому обычно в течение нескольких секунд последовательно поступает несколько декодов с разными уровнями; из них выбирается лучший.

Строка декода, выводимая на экран и записываемая в файл, содержит дату, время, позывной, частоту, QRB (если известен qth), оценку битового отношения сигнал-шум на выходе декодера Eb/No, отношение сигнал-шум SNR (вычисляется отношение данного спектрального максимума к среднему значению амплитуд спектра по всей полосе 100 Гц; чтобы результат как-то соотносился с результатами Opera, к найденной величине прибавляется поправка, задаваемая параметром

OPERA_DB_OFFSET=-10 – для слабых сигналов это примерно -10 дБ). Выводится количество итераций декодера (1-декодировано сразу правильно, меньше 83 – с коротким списком, больше 83 – с длинным списком; если декод обнаружен битовым коррелятором, то выводится коэффицент корреляции в процентах; для мягкого коррелятора поле остается пустым).
Следующее поле флагов заполняется так:
а) если совпала только одна CRC, выводится знак ‘?’;
б) если декодированный позывной отсутствует в списке известных позывных, выводится знак ‘-’;
в) если декод получен битовым коррелятором выводится символ ‘c’;
г) для opds ставится ‘d’.
д) для достоверного декода ничего дополнительно не выводится (‘ ‘).

Практика показала, что (в версии 1.7.0) чувствительность мягкого коррелятора лишь ненамного превышает чувствительность битового коррелятора, что оказалось несколько неожиданным. Этот вопрос еще будет исследован в дальнейшем, однако пока достичь чувствительности к слабым сигналам OPDS by DF6NM не удается. Зато программа лучше декодирует дрейфующие по частоте сигналы.

В целом же, применение корреляторов позволяет правильно декодировть сигнал при SNR ниже -40 дБ. Нижней границей пока можно считать уровень -43…-47 дБ в полосе 1 кГц. Это соответствует сигналу, который уже не читается на водопаде и отмечаются лишь признаки присутствия на спектрограмме в режиме QRSS-3.

Когда режим «мягкого» коррелятора включен, дополнительно один раз в 10-15 минут осуществляется «обзор» с поиском возможно передаваемых сигналов. Результаты обзора, позывной, набравший наибольшую корреляционную сумму по всем каналам и всем временным сдвигам, выводится в строку состояния с указанием ожидаемого времени окончания его передачи (если это время меньше текущего, значит сигнал передавался ранее и сейчас в памяти присутствует его «хвостик»).

Когда зафиксирован декод, помимо сохранения скриншота с уникальным именем в указанную дерикторию (CAPTURE_PATH=Capture\), осуществляется сохранение этого же скриншота в файл с постоянным именем (CAPTURE_FILENAME=op32rx_capture.png или .bmp, .jpg). Это удобно для последующей отправки скриншота на сайт граббера. Для этой же цели может служить и какая-либо команда, автоматически выполняемая после фиксации декода (EXECUTE_IF_DECODED_ENABLE=1 EXECUTE_IF_DECODED=cmd /C start /MIN file_upload.bat). Строка команды может быть произвольной в формате командного интерпретатора cmd.exe. Чтобы консольное окно, в котором будет выполняться file_upload.bat, закрылось по окончании исполнения, не забудьте в конце файла добавить команду exit. Ввод и тестирование команды можно осуществить в окне дополнительных настроек.

Ниже приведена структура ini-файла с комментариями:

```
[OP-32]
мой позывной
MYCALL=RN3AUS

мой локатор
MYQTH=KO85FN

заголовок для файла
MYHEADER=RN3AUS KO85FN Opera-32 decoded (Op32Rx RN3AUS software)

Центральная частота аудио спектра (от 500 до 23500)
F_CENTER=1500
```

```
Частота настройки приемника (произвольно)
F_DIAL=136000

Минимальное расстояние между спектральными максимумами (один шаг равен 0.122 Гц)
Рекомендуемое значение 4 или 5
MIN_PEAK_SPACE=4

Количество спектральных максимумов ( -1 – авто). Рекомендуемое значение 20…30.
N_MAX=20

Минимальное превышение пика над соседним уровнем, дБ (0-нет порога).
Рекомендуется 0,4…0,8 – так чтобы в результате число пиков получалось 10-30
PEAK_THRESHOLD=0,4

файл для записи спотов
FILE=my_detected.txt

разрешить сохранять скриншоты в момент декода (только для GUI версии)
CAPTURE_ENABLE=1

путь для сохранения скриншотов (только для GUI версии)
CAPTURE_PATH=Capture\

дополнительно файл с постоянным именем для сохранения скриншотов
CAPTURE_FILENAME=op32rx_capture.png


игнорировать частоты из списка loran_lines.txt (1-да 0-нет)
IGNORE_LORAN_LINES=0

всегда анализировать частоты из списка freq_list.txt (1-да 0-нет)
USE_FREQUENCY_LIST=1

отсеивать незнакомые позывные, отсутствующие в callsloc.txt и known_list.txt
Рекомендуется всегда включать этот режим
USE_KNOWN_CALLSLIST=1

использовать декодер с глубоким поиском (короткий список). Рекомендуется.
DEEP_DECODER=1

использовать декодер с длинным списком. Обычно это не требуется.
ULTRA_DEEP_DECODER=0

использовать когерентный детектор (0-амплитудный квадратичный, 1 - когеррентный)
USE_COHERENT_DETECTOR=0

сдвиг оценок SNR (dB) для единообразия с Opera, соответствует SNR в полосе 1 кГц
OPERA_DB_OFFSET=-10

порог (dB) декодера (рекомендуется 3…4)
OP_THRESHOLD=3,5

порог (dB) выхода корреляторов (рекомендуется 0,2 … 2,5; следует подобрать
мнимальный порог, исключающий ложные обнаружения)
OPDS_THRESHOLD=0,5

порог битового коррелятора (количество совпадающих бит 159…163). Нужно подобрать
минимальный порог, обеспечивающий отсутствие ложных декодов.
CORRELATOR_237_THRESHOLD=161
```

```
минимальное превышение наибольшего корреляционного пика над остальными (в разах)
OPDS_MAX1MAX2_THRESHOLD=1,6

использовать битовый коррелятор (0-нет 1-да)
USE_CORRELATOR=1

использовать мягкий коррелятор opds (0-нет 1-да)
USE_OPDS=0

декодер проверяет 4 младших бита декодированной информации, которые всегда
нулевые (0-нет, 1-да). Если биты не нулевые, декод помечается знаком ?
DECODER_ZERO_TEST=1

время блокировки позывного от повторного срабатывания (сек) 0…1800
TIME_CALL_BLOCKED=600

отслеживать частотный дрейф (0-нет, 1-да). Рекомендуется включать.
DECODER_DRIFT_TRACKING=1

компенсировать различие тактовых частот (0-нет, 1-да). Рекомендуется включать.
DECODER_SR_COMPENSATE=1

Отображать осциллограмму (0-нет 1-да)
OSCILLOGRAM_VISIBLE=1

Отображать спектр (0-нет 1-да)
SPECTRUM_VISIBLE=1

Контрастность водопада (0-малая 1-средняя 2-высокая) рекомендуется 0
WATERFALL_CONTRAST=0

Режим отображения в поле осциллограммы (0-осциилограмма 1-выход детектора 2-
полоска спектрограммы в режиме QRSS-3)
OSCILL_MODE=1

выполнить команду в момент обнаружения декода (0-нет 1-да)
EXECUTE_IF_DECODED_ENABLE=0

текст команды в формате команд cmd.exe (например: cmd /C start /MIN
file_upload.bat) без кавычек.
EXECUTE_IF_DECODED=

Записывать споты в файл в обратном порядке (вверху последний по времени) (0-нет
1-да). Рекомендуется 1.
REVERSE_OUTPUT=1

[SOUND]
используемая звуковая карта (-1 по умолчанию)
DEVICE=-1

[PSKREPORTER]
HOSTNAME=report.pskreporter.info
PORT=4739

разрешить отправлять споты на сервер (0-нет 1-да)
UPLOAD_SPOT=1
```

разрешить отправку спотов коррелятора (0-нет 1-да). Включайте, если уверены в малом количестве ложных декодов коррелятора
UPLOAD_OPDS_SPOT=0

Порядок установки и работы с программой ничем не отличается от версии 1.0.0 (смотри соответствующий хелп).