4. РАЗРАБОТКА ПРОГРАМНЫХ МОДУЛЕЙ

4.1 Описание алгоритмов

Ядром программы является тема обработки звука, поэтому рассмотрены будут алгоритмы, связанные с этой задачей. Для наглядного представления о том, как работает механизм распознавания частоты, можно ознакомиться с диаграммой последовательности программы (ПРИЛОЖЕНИЕ В), где так же приведена последовательность отображения результата.

Сразу после инициализации микрофона запускается процесс записи поступающих звуковых сигналов. Для анализа частот требуется лишь время от времени считывать данные из внутреннего буфера микрофона. После получения ИКМ, она приводится к типу double для упрощения вычислений и начинается процесс обработки: сначала к ИКМ прикладывается оконная функция, для устранения шумов, затем начинается этап расчета спектрограммы с помощью БПФ. Для этих целей был выбран алгоритм БПФ по Кули-Тьюки, в связи с его высокой эффективностью. Итеративная реализация алгоритма была найдена в виде псевдокода, после чего была переведена на Java. На выходе преобразований получается массив данных, где каждый индекс соответствует некоторому дискретному значению частоты, а само значение массива – амплитуде этой частоты. Для того чтобы увеличить точно расчета частоты, начинается поиск не одного пикового значения массива, а их группы, после чего начинается анализ полученной группы пиков. В конце концов, на выходе имеется значение частоты, рассчитанное с намного большей точностью, нежели чем при анализе одного пика.

4.2 Описание алгоритма по шагам

Рассмотрим подробнее алгоритм анализа группы пиков поступающего сигнала – метод scanSignalIntervals. Схема алгоритма приведена в приложениях (ПРИЛОЖЕНИЕ Г). Код алгоритма приведен в ПРИЛОЖЕНИИ Д.

Используемые переменные:

1. x – массив входных данных ИКМ
2. index – начальный индекс для работы с данными массива
3. length – длина обрабатываемого фрагмента
4. intervalMin – минимальное значение периода на данном этапе
5. intervalMax – максимальное значение периода на данном этапе
6. diff – разность двух значений амплитуд из ИКМ, отстоящих друг от друга на некоторый период
7. optimalValue – оптимальное значение суммы разностей сигналов
8. optimalInterval – оптимальное значение периода

9. steps – число шагов между рассматриваемыми периодами
10. maxAmountOfSteps – ограничитель числа шагов

Описание алгоритма нахождения оптимального периода:

Шаг 1: Начало.

Шаг 2: Задание входных параметров: x, index, length, intervalMin, intervalMax.

Шаг 3: Задание максимального числа шагов maxAmountOfSteps.

Шаг 4: Расчет числа шагов между крайними значениями периодов.

Шаг 5: Если полученное число шагов больше максимального допустимого, то ограничить его последним, иначе переход на следующий Шаг с текущим значением steps.

Шаг 6: Если число шагов меньше 0, то присвоить последнему единицу, иначе перейти на следующий Шаг с текущим значением steps.

Шаг 7: Цикл i от 0 до steps, чтобы рассмотреть все дискретные периоды между заданными

Шаг 8: Рассчитать текущее рассматриваемое значение периода.

Шаг 9: Цикл j от 0 до length, чтобы пройти по рассматриваемому фрагменту массива

Шаг 10: Рассчитать квадрат разницы между значениями входной ИКМ, отстоящими друг от друга на текущий период.

Шаг 11: Нарастить j. Если j становится равным length перейти на Шаг 12, иначе перейти на Шаг 9.

Шаг 12: Если оптимальное значение больше рассчитанной суммы, то перейти на Шаг 13, иначе перейти на Шаг 14.

Шаг 13: Присвоить оптимальному значению суммы новую сумму, а оптимальному значению периода, рассматриваемый период.

Шаг 14: Нарастить i. Если i становится равным steps, то перейти на Шаг 15, иначе перейти на Шаг 7.

Шаг 15: Конец.